1990-12

Including state excitation in the fixed-interval smoothing algorithm and implementation of the maneuver detection method using error residuals

Sevim, Alaettin

Monterey, California: Naval Postgraduate School

http://hdl.handle.net/10945/27667

D-A246 336

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

INCLUDING STATE EXCITATION IN THE
FIXED-INTERVAL SMOOTHING
ALGORITHM AND IMPLEMENTATION OF THE
MANEUVER DETECTION METHOD
USING ERROR RESIDUALS

by

Alaettin Sevim

December 1990

Thesis Advisor   Harold A. Titus

92-04574

92 2 21 020

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified		1b Restrictive Markings			
2a Security Classification Authority		3 Distribution/Availability of Report			
2b Declassification/Downgrading Schedule		Approved for public release; distribution is unlimited.			
4 Performing Organization Report Number(s)		5 Monitoring Organization Report Number(s)			
6a Name of Performing Organization Naval Postgraduate School	6b Office Symbol (if applicable) XX	7a Name of Monitoring Organization Naval Postgraduate School			
6c Address (city, state, and ZIP code) Monterey, CA 93943-5000		7b Address (city, state, and ZIP code) Monterey, CA 93943-5000			
8a Name of Funding/Sponsoring Organization	8b Office Symbol (if applicable)	9 Procurement Instrument Identification Number			
8c Address (city, state, and ZIP code)		10 Source of Funding Numbers			
		Program Element No	Project No	Task No	Work Unit Accession No

11 Title (Include security classification) INCLUDING STATE EXCITATION IN THE FIXED-INTERVAL SMOOTHING ALGORITHM AND IMPLEMENTATION OF THE MANEUVER DETECTION METHOD USING ERROR RESIDUALS
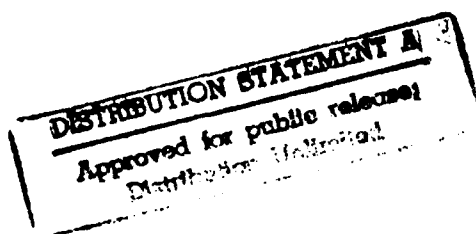
12 Personal Author(s) Alaettin Sevim

13a Type of Report Master's Thesis	13b Time Covered From To	14 Date of Report (year, month, day) December 1990	15 Page Count 112

16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms (continue on reverse if necessary and identify by block number)
Field	Group	Subgroup	Kalman Filter, Smoothing, Noise Process, Maneuver Detection.

19 Abstract (continue on reverse if necessary and identify by block number)

The effects of the state excitation matrix Q_K in the smoothing routine of an extended Kalman filter is investigated. A new algorithm to derive the Q_K matrix is also developed. In addition, the accuracy of the filter was substantially improved by implementing a new maneuver detection technique. Several tracking scenarios are simulated and analyzed for noise free and noisy cases and statistical data are obtained for the maneuver detection technique. The program codes are included as appendices.

20 Distribution/Availability of Abstract ☒ unclassified/unlimited ☐ same as report ☐ DTIC users	21 Abstract Security Classification Unclassified	
22a Name of Responsible Individual Harold A. Titus	22b Telephone (include Area code) (408) 646-2560	22c Office Symbol 62TS

DD FORM 1473,84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

security classification of this page

Unclassified

Including State Excitation in the Fixed-Interval Smoothing
Algorithm and Implementation of the Maneuver Detection Method
Using Error Residuals

by

Alaettin Sevim
Lieutenant Junior Grade, Turkish Navy
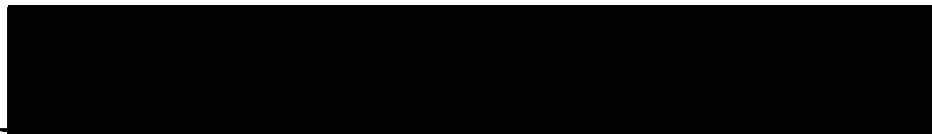B.S., Turkish Naval Academy, 1984

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE

from the
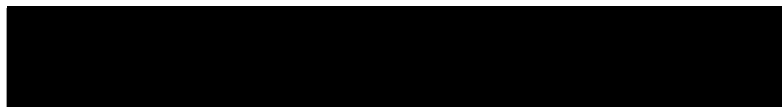
NAVAL POSTGRADUATE SCHOOL
December 1990
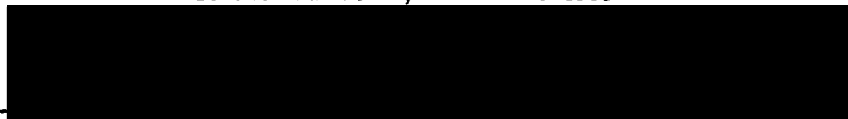
Author: _____
Alaettin Sevim

Approved by: _____
Harold A. Titus, Thesis Advisor

_____
Jeff Burl, Second Reader

_____
Michael A. Morgan, Chairman,
Department of Electrical and Computer Engineering

ii

ABSTRACT

The effects of the state excitation matrix Q_K in the smoothing routine of an extended Kalman filter is investigated. A new algorithm to derive the Q_K matrix is also developed. In addition, the accuracy of the filter was substantially improved by implementing a new maneuver detection technique. Several tracking scenarios are simulated and analyzed for noise free and noisy cases and statistical data are obtained for the maneuver detection technique. The program codes are included as appendices.

Accession For

NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	

By

Distribution/

Availability Codes

| Dist | Avail and/or Special |
| A-1 | |

DTIC COPY INSPECTED 4

iii

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I. INTRODUCTION

During the last century, man has reached out over ever-increasing distances. Manmade devices have been sent beyond our solar system and to the deepest points of the oceans. These recent developments have focused new attention on an existing problem: how to accurately track long-range devices along their voyages in unknown environments. This problem is made even worse when only passive sensors can be used. One particular problem applicable to naval technology is tracking a ship by lines of bearing obtained by passive sensors. A powerful method of dealing with this problem, known as Kalman filtering, has been used with great success since Kalman and Bucy [Refs. 1, 2] first presented its principles 30 years ago.

This report further develops an existing Kalman filter to which a fixed-interval smoothing algorithm has been added. In this research, we examine how the overall accuracy of the extended Kalman filter is affected by applying a noise process in the smoothing algorithm. We also develop a new maneuver detection technique and study how the filter performs when using it. This research is based on previous work done by Lieutenant Thomas K. Bennett [Ref. 3] and Lieutenant William J. Galinis [Ref. 4]. They investigated the problems of two ships tracking a third only by passive radio direction finding equipment.

This report is organized into six major sections. The first section is this introduction, which serves as a guide to approaching this report. In Chapter II, the physical tracking system used for simulations in this report is modeled. Chapter III gives the basic principles of the Kalman filtering and fixed-interval smoothing. In Chapter IV, we investigate how the noise process in the smoothing routine and a new maneuver detection technique affect the accuracy of the extended Kalman filter. Chapters V and VI

1

show the simulations and present the conclusions. The appendices list the program codes used in this research.

PROBLEM STATEMENT

A. THE SYSTEM MOL ᴸ

The system used in this thesis includes two sensors and one target ship. A two-dimensional cartesian coordinate system is used, in which the positive x and positive y directions correspond to East and North, respectively. The target and sensor ships are both free to move throughout this coordinate space. For simplicity, the following assumptions are made during the development of this model: [Ref. 4]

- The effect of the wind, current and other forces on the ship are negligible.
- The ocean surface is considered flat; the curvature of the earth is neglected.
- Course and speed inputs are taken as constants (i.e., step inputs).

From [Refs. 5: p. 168,6: pp.12-13], the discrete-time, state-space representation ᵕf the model described above is

$$\hat{x}_{K+1} = \phi_K \hat{x}_K + \hat{\omega}_K \tag{2.1}$$

where

$\hat{x}_{K+1}$ = state estimate vector,

$\hat{x}_K$ = state vector,

ϕ_K = state transition matrix and

$\hat{\omega}_K$ = disturbance.

A state vector $\hat{x}_K$ is defined to contain the minimum number of the elements necessary to describe the target. A fourth order state vector for this model, then, consists of the position and velocity of the target in both x and y directions.

$$\hat{x}_K = \begin{bmatrix} x_K \\ \dot{x}_K \\ y_K \\ \dot{y}_K \end{bmatrix} \qquad (2.2)$$

Next, a state transition matrix ϕ_K is chosen to fit the target dynamics. Since the target modeled in this problem moves linearly at a constant velocity, the ϕ_K matrix is

$$\phi_K = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.3)$$

where T is the observation interval.

The unpredictable accelerations of the target are taken into account using the noise vector $\hat{\omega}_K$. The noise vector is a function of the transition matrix Γ_K and the acceleration matrix a_K:

$$\hat{\omega}_K = \Gamma_K a_K = \Gamma_K \begin{bmatrix} a_{x_K} \\ a_{y_K} \end{bmatrix} \qquad (2.4)$$

where the noise transition matrix Γ_K is defined as

$$\Gamma_K = \begin{bmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{bmatrix} \qquad (2.5)$$

4

Putting Equations (2.2) through (2.5) into Equation (2.1), the final state-space equation for the system modeled in this problem can be written as

$$
\begin{bmatrix} x_{K+1} \\ \dot{x}_{K+1} \\ y_{K+1} \\ \dot{y}_{K+1} \end{bmatrix} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_K \\ \dot{x}_K \\ y_K \\ \dot{y}_K \end{bmatrix} + \begin{bmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{bmatrix} \begin{bmatrix} a_{x_K} \\ a_{y_K} \end{bmatrix} \tag{2.6}
$$

B. THE MEASUREMENT MODEL

For linear systems, measurements can be modeled using the following linear measurement equation. [Refs. 5: p. 168,6: pp. 12-13]

$$
\hat{z}_{K+1} = H\hat{x}_{K+1} + \hat{\mu}_{K+1} \tag{2.7}
$$

where

$\hat{z}_{K+1}$ = measurements,

H = observation matrix,

$\hat{x}_{K+1}$ = state estimate vector and

$\hat{\mu}_{K+1}$ = measurement noise.

Unfortunately, many real systems are not linear. The system we studied in this thesis falls in this category. Although this system has a linear state-transition equation, it has a non-linear measurement equation, since the measurements, lines of bearings, are non-linear functions of the system states. As it can be seen from the geometry of the typical scenario in Figure 1, an appropriate model with measurement noise included for the non-linear measurement process of this system would instead be [Ref. 3]

$$
\hat{z}_{n_K} = \tan^{-1}\left[\frac{x_{t_K} - x_{n_K}}{y_{t_K} - y_{n_K}} \right] + \hat{\mu}_K \tag{2.8}
$$

5
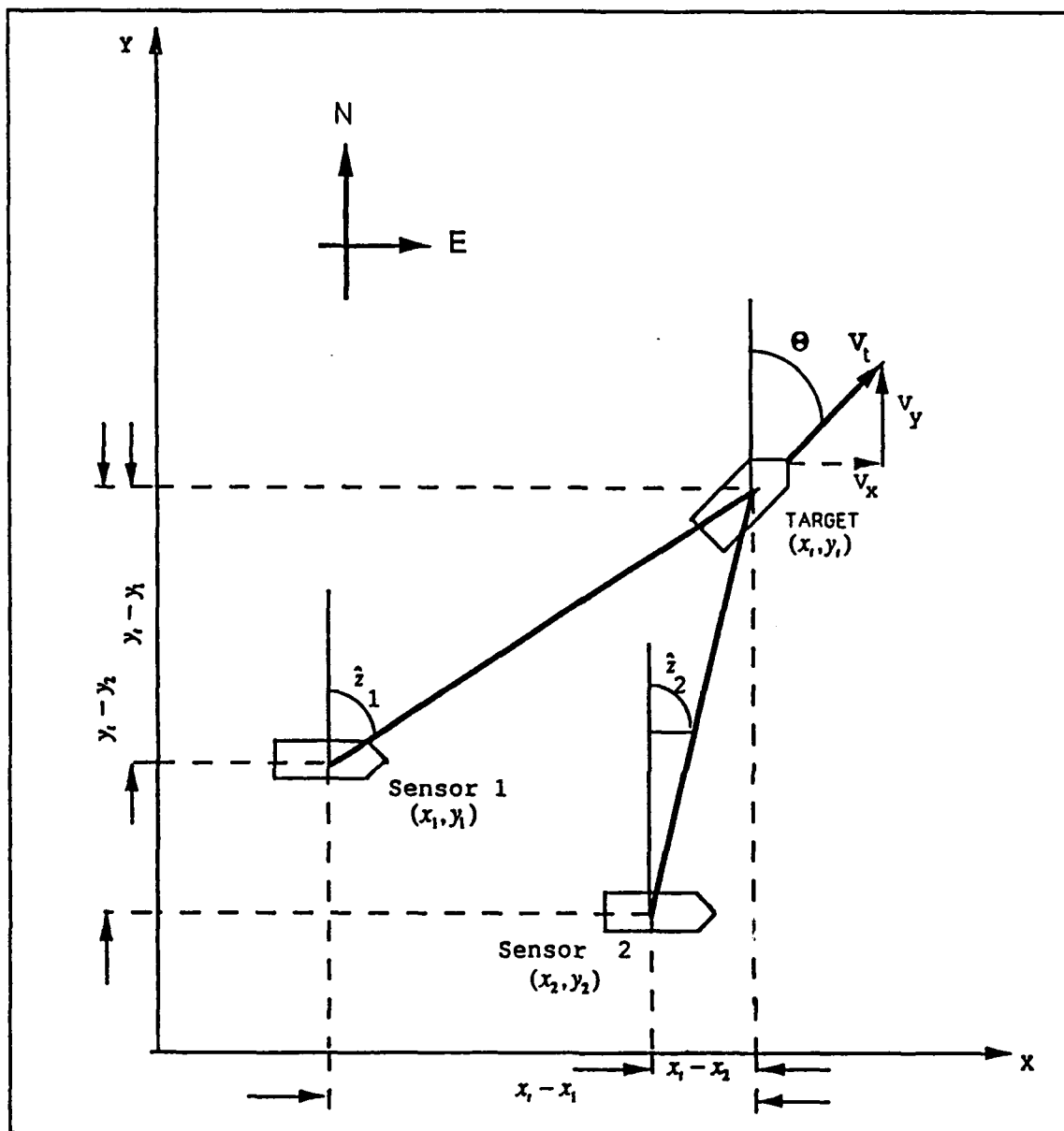
Figure 1. **Typical Tracking Scenario**

where

$\hat{z}_{n_K}$ = observed lines of bearing by a sensor ship n, at time k,

x_{t_K} , y_{t_K} = position of the target ship on x, y axes, at time k,

x_{n_K} , y_{n_K} = position of the sensor ship n on x, y axes, at time k and

6

$\hat{\mu}_K$ = measurement noise.

Although there are several types of noise which disturb the measurements, it is the atmospheric noise that makes the major contribution in the frequency range of interest in this study. This is generally a non-white, non-Gaussian process. However, it can be considered to be a white Gaussian process over an extended period of time in order to more easily implement the extended Kalman filter. In this application, a white noise model is used for the study of noisy cases.

III. THEORY

A. KALMAN FILTER

The Kalman filter removes random noise from the state estimates of a system by adding a weighted error term to the predicted state estimates. The error term is simply the difference between the filter's prediction of the measurement and the observed value of that measurement at a particular time. The weighting factor, also called the filter gain, is based on the predicted covariance of error between estimates and observed values. The basic operation of the filter can be described in several steps:

A priori estimates of the state $\hat{x}_{K/K}$ are projected in time to some predicted state estimate $\hat{x}_{K+1/K}$, and the predicted error covariance $P_{K+1/K}$ of these estimates is calculated. The filter then calculates a gain vector G_{K+1}, based on the predicted error covariance. As mentioned before, the error is the difference between observed and predicted measurements. Next, this error is multiplied by the filter gain and the result is added to the predicted state estimates to give the updated estimate $\hat{x}_{K+1/K+1}$. The updated value of error covariance $P_{K+1/K+1}$ is also calculated.

In short, the Kalman filter is a linear, minimum variance estimator. A block diagram of the filter is in Figure 2. A more detailed explanation of the filter's operation will be given later in this chapter. For further information on the derivation and application of the Kalman filter, the reader should refer to [Refs. 7,8,9]

B. EXTENDED KALMAN FILTER

The Kalman filter explained above calculates the optimal estimate for the states of linear systems. As mentioned before, the system we studied in this thesis has a linear state-transition equation and non-linear measurement equation. Therefore it is not lin-

8

A Priori Information
━━━━━━━━━━━━━━━━
- Initial State Estimate
- Initial Error Covariance
- State Transition Matrix
- Measurement Noise Covariance

Observations $\hat{z}_{K+1}$

KALMAN FILTER

State Estimate $\hat{X}_{K+1/K+1}$

Error Estimate $P_{K+1/K+1}$
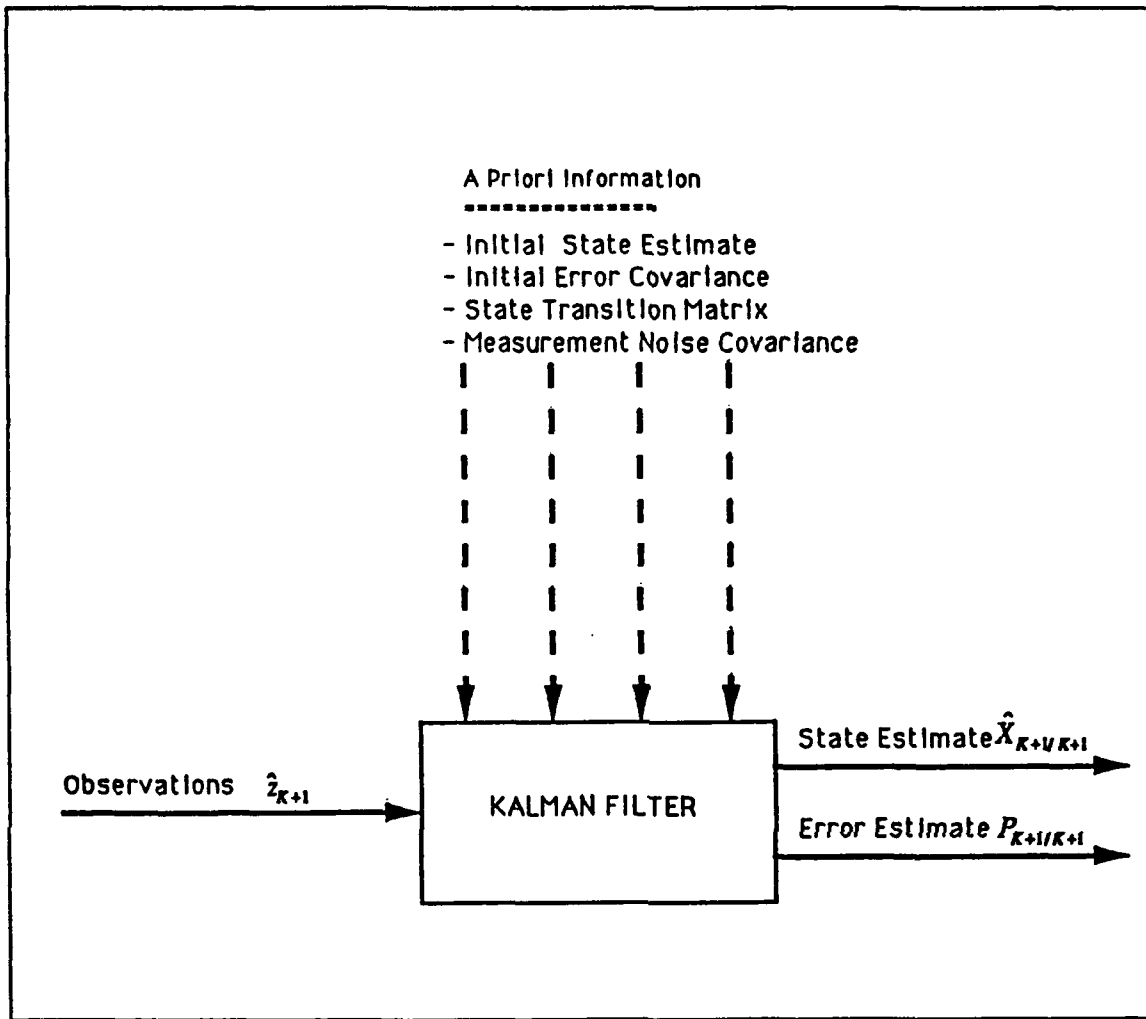
Figure 2.   Block Diagram of the Kalman Filter.

ear. The adaptation of the Kalman filter to a non-linear system is called the extended Kalman filter.

For the system studied in this thesis, the non-linear measurement equation can be defined as:

$$z_{K+1} = H(\hat{x}_{K+1}) + \mu_{K+1} \tag{3.1}$$

9

We see that the only difference between this equation and the linear measurement equation (2.7) is the observation matrix H. The H matrix is now a function of the system states. In order to linearize the measurement equation, we have chosen in this thesis to expand the observation matrix H in a Taylor series around the current estimate and then to use only the first order term.

The following linearized form of the measurement equation is obtained directly from previous work on this subject. Its development will not be repeated here since an excellent derivation of it is presented in these reports [Refs. 3,4]. The equations are:

$$H_{K+1} = \left[\frac{\hat{y}_{t_{K+1/K}} - y_{n_{K+1}}}{\hat{R}^2_{K+1}} \quad 0 \quad -\frac{\hat{x}_{t_{K+1/K}} - x_{n_{K+1}}}{\hat{R}^2_{K+1}} \quad 0 \right] \tag{3.2}$$

and

$$\hat{R}^2_{K+1} = (\hat{y}_{t_{K+1/K}} - y_{n_{K+1}})^2 + (\hat{x}_{t_{K+1/K}} - x_{n_{K+1}})^2$$

where

$\hat{x}_{t_{K+1/K}}, \hat{y}_{t_{K+1/K}} =$ The position estimates of the target at time $K+1$, based on the previous value at time K.

$x_{n_{K+1}}, y_{n_{K+1}} =$ The position of the sensor ship n at time $K+1$.

Once the measurement process is linearized, the normal linear Kalman filter equations can be used in the estimation process. The following Kalman filter equations, taken from [Ref. 5] and derived in [Refs. 5,10], are:

$$\hat{x}_{K+1/K} = \phi \hat{x}_{K/K} \tag{3.3}$$

$$P_{K+1/K} = \phi P_{K/K} \phi^T + Q_{K+1} \tag{3.4}$$

$$G_{K+1} = P_{K+1/K} H_{K+1}^T \left[H_{K+1} P_{K+1/K} H_{K+1}^T + R \right]^{-1} \tag{3.5}$$

$$\hat{x}_{K+1/K+1} = \hat{x}_{K+1/K} + G_{K+1} \left[z_{K+1} - H_{K+1} \hat{x}_{K+1/K} \right] \tag{3.6}$$

$$P_{K+1/K+1} = \left[I - G_{K+1} H_{K+1} \right] P_{K+1/K} \tag{3.7}$$

The variables are defined as follows:

$\hat{x}_{K+1/K}$ = predicted state estimate,

$\hat{x}_{K/K}$ = state estimate (state vector),

ϕ = state transation matrix given by equation (2.3),

$P_{K+1/K}$ = predicted state error covariance,

$P_{K/K}$ = state error covariance,

Q_{K+1} = state excitation matrix,

G_{K+1} = Kalman gain matrix,

R = measurement noise covariance matrix and

H_{K+1} = linearized measurement matrix given by equation (3.2).

The measurement noise covariance matrix R is a indication of the accuracy of the measurements made. This matrix is:

$$R = \left[\mu_K \mu_K^T \right] \tag{3.8}$$

The state excitation matrix Q_{K+1} used in equation (3.4) represents the system noise process. This term is a measure of how closely the system model actually represents the real system and to what degree the system is affected by noise. The derivation of the Q_{K+1} matrix will be studied in the next chapter.

As can be seen from equations (3.3) through (3.7), the basic operation of the filter is a relatively straightforward recursive process. But the filter must be initialized before

11

processing the measurement data. When the filter is initialized, no prior value for the state estimate $\hat{x}_{K/K}$ exists. Therefore the value of the first observed position is assigned to it. The coordinates of observed positions can be calculated from the two lines of bearings by using the following equations:

$$x_t = \left[-\frac{y_2 \tan(\theta_2) + y_1 \tan(\theta_1) + x_2 - x_1}{\tan(\theta_1) - \tan(\theta_2)} - y_1 \right] \tan(\theta_1) + x_1 \qquad (3.9)$$

$$y_t = \left[-\frac{y_2 \tan(\theta_2) + y_1 \tan(\theta_1) + x_2 - x_1}{\tan(\theta_1) - \tan(\theta_2)} \right] \qquad (3.10)$$

Since there is no prior velocity information available at the moment of initialization, the initial velocity estimate is taken as zero. Figure 3 shows the initialization procedure.

Since the initial state estimates will have some error, we pick some starting values for the errors in initial position and velocity to initialize the error covariance matrix. These are 100 nautical miles (Nm) in position and 0.5 Nm per minute (i.e., 30 kts) for velocity [Refs. 3,4]. The error covariance matrix can now be initialized as:

$$P_{0/-1} = \begin{bmatrix} 10000 & 0 & 0 & 0 \\ 0 & 0.25 & 0 & 0 \\ 0 & 0 & 10000 & 0 \\ 0 & 0 & 0 & 0.25 \end{bmatrix} \qquad (3.11)$$

Once initialized, the filter is ready to process the measurements. First, the state estimate and state error covariance matrixes are projected to the present time using the ϕ matrix. Next, these predicted state estimates are used to calculate the H matrix. Finally, the Kalman gains are calculated. The Kalman gain is a measure of where the filter's confidence is being placed: either in the filter's estimation or in the current observation. As is se in equation (3.5) the value of the Kalman gain is based on the predicted error

12
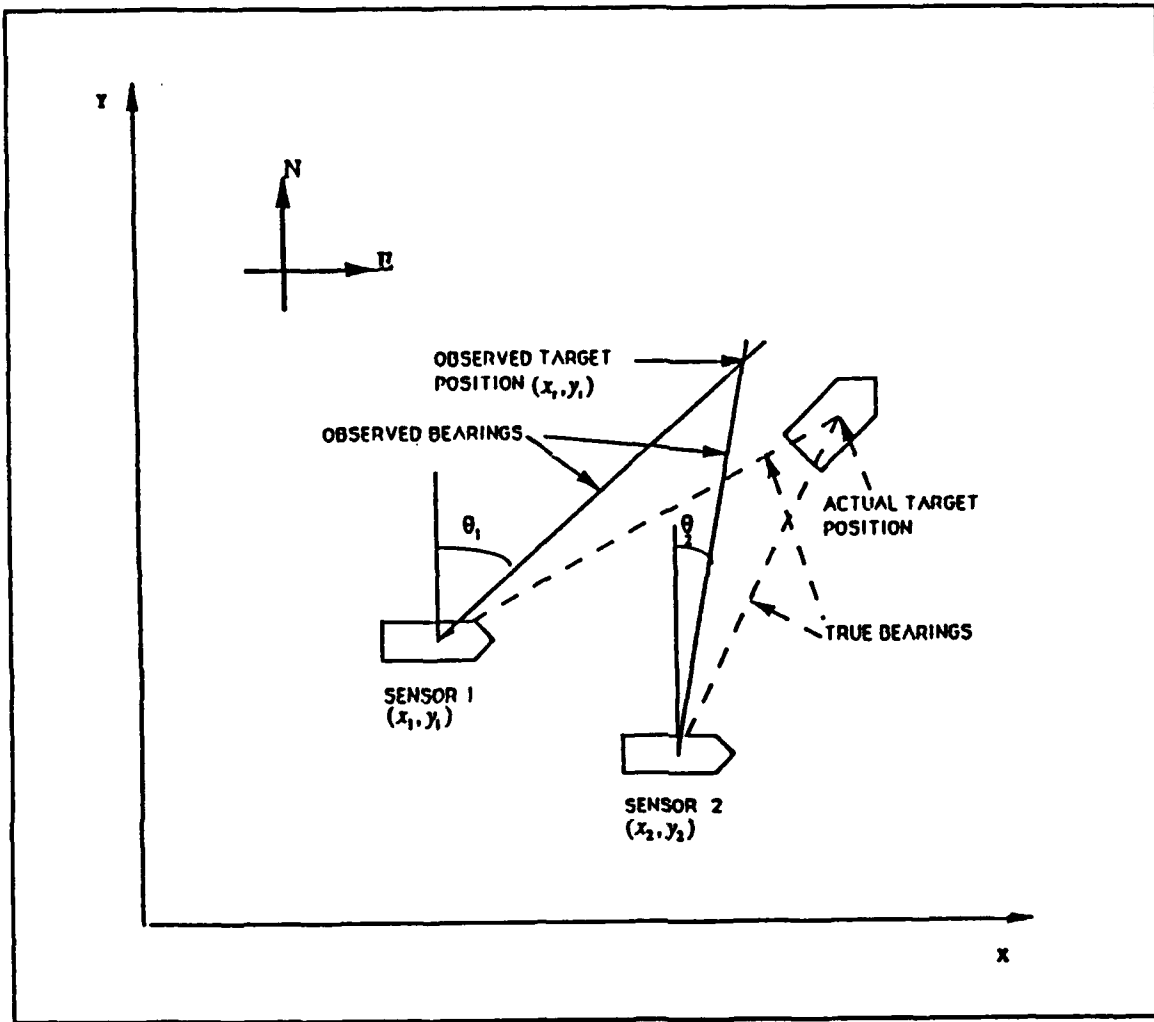
Figure 3. The Initialization Procedure.

covariance matrix. If $P_{K+1/K}$ is large the Kalman gain will approach unity. If $P_{K+1/K}$ is small, the gain will approach zero due to the finite value of the measurement noise covariance R. By manipulating equation (3.6) we can see how varying the Kalman gain affects the process of updating state estimates.

$$\hat{x}_{K+1/K+1} = [I - G_{K+1}H_{K+1}]\hat{x}_{K+1/K} + G_{K+1}z_{K+1} \tag{3.12}$$

13

As mentioned before, this equation shows that a large Kalman gain places more weight on the current observation. On the other hand, a small gain causes the factor of $[I - G_{K+1}H_{K+1}]$ in equation (3.12) to approach unity, in this sense placing more emphasis on the filter's estimates. As can be seen from equation (3.7), the factor of $[I - G_{K+1}K_{K+1}]$ is used to update state error covariance matrix.

C. SMOOTHING ALGORITHM

Smoothing is a non-real-time data process used to reduce error in state estimates produced by a Kalman filter. Let time K be within the time interval 0 to N, so that $0 \leq K \leq N$. A Kalman filter's state estimate for time K, denoted by $\hat{x}_{K/K}$, is based only on measurements occuring up to time K. But the smoothed state estimate is based on the measurements that occurred over the entire time interval 0 to N. This smoothed estimate is denoted by $\hat{x}_{K/N}$. The smoothed error covariance at time K is represented by $P_{K/N}$. This quantity has no impact on the calculation of the smoothed estimate $\hat{x}_{K/N}$ but it is an indicator of how well the smoothing filter is working. If $P_{K/N} \leq P_{K/K}$, the smoothed estimate is better than or equal to its filtered estimate except for the last data point where both smoothed and filtered estimates are equal. The smoothing algorithm operates backwards in time, beginning at time N and ending at time zero. Therefore, since the last filtered estimate at time N is taken as the first smoothed estimate, $P_{K/N}$ must be equal to $P_{K/K}$ at this last data point. This can be seen graphically in Figure 4.

Meditch [Ref. 5] places smoothed estimates into three classes:

Fixed-Interval smoothed estimate , denoted by $\hat{x}_{K/N}$ where K = 0, 1, ..., N-1; N is a positive integer.

Fixed-Point smoothed estimate , denoted by $\hat{x}_{K/J}$ where J = K+1, K+2, ...; K is a fixed integer.

Fixed-Lag smoothed estimate , denoted by $\hat{x}_{K/K+N}$ where K = 0, 1, ...; N is a fixed positive integer.

14
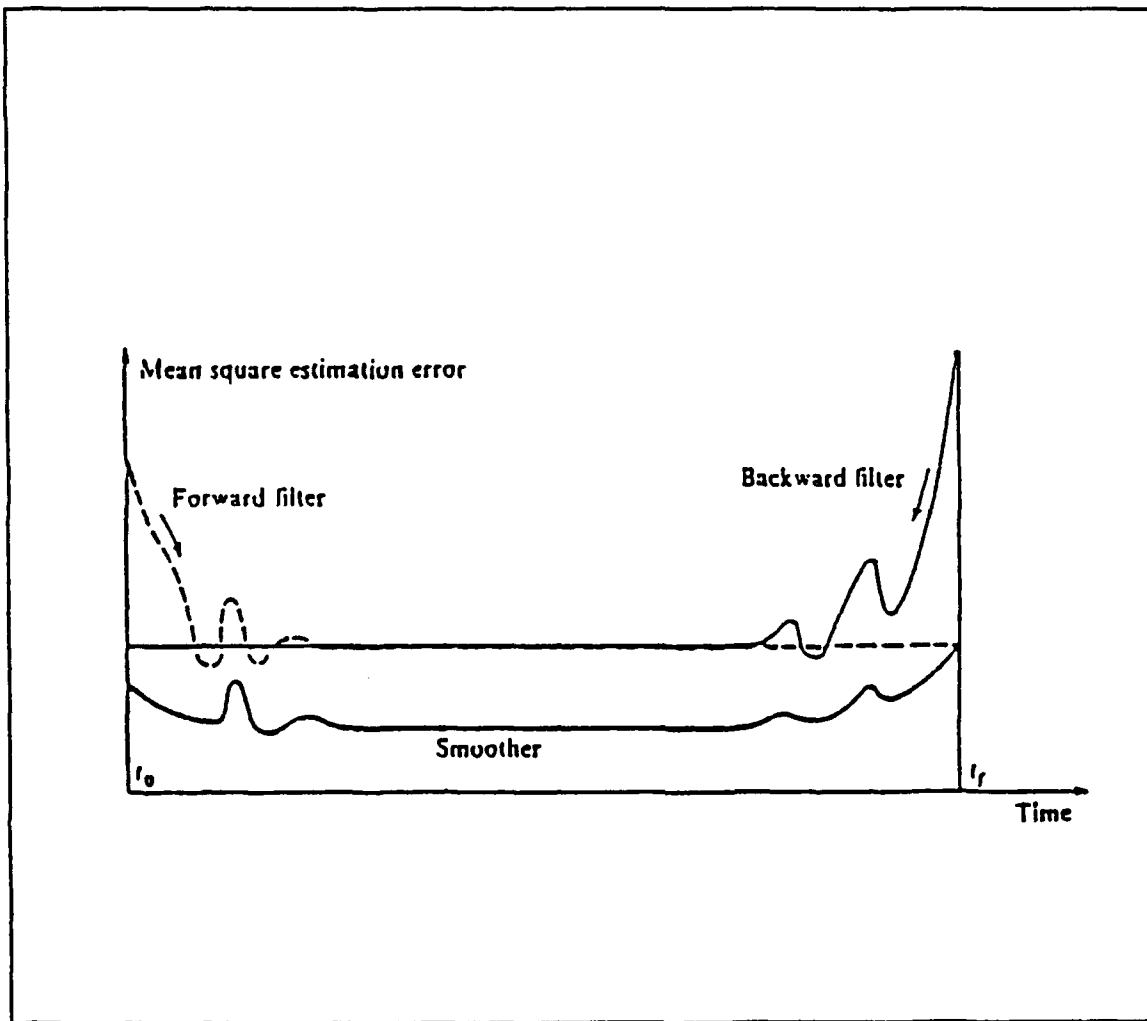
Figure 4. Advantage of the Performing Optimal Smoothing.

In this thesis a fixed-interval smoothing filter is used. The basic block diagram of this filter is shown in Figure 5. The equations to implement the smoothing algorithm are obtained The equations to implement the smoothing algorithm are obtained from [Ref. 5: pp. 216-224] and are shown below:

$$A_K = P_{K/K}\phi^T P^{-1}_{K+1/K}$$ (3.13)

$$\hat{x}_{K/N} = \hat{x}_{K/K} + A_K[\hat{x}_{K+1/N} - \hat{x}_{K+1/K}]$$ (3.14)

15

FROM EXTENDED KALMAN FILTER

$\hat{x}_{K+1/K}$ $\hat{x}_{K/K}$

Λ_K

SMOOTHED STATE
ESTIMATE $\hat{x}_{K/N}$

$\hat{x}_{K+1/N}$

DELAY

$K = N-1,...,0$
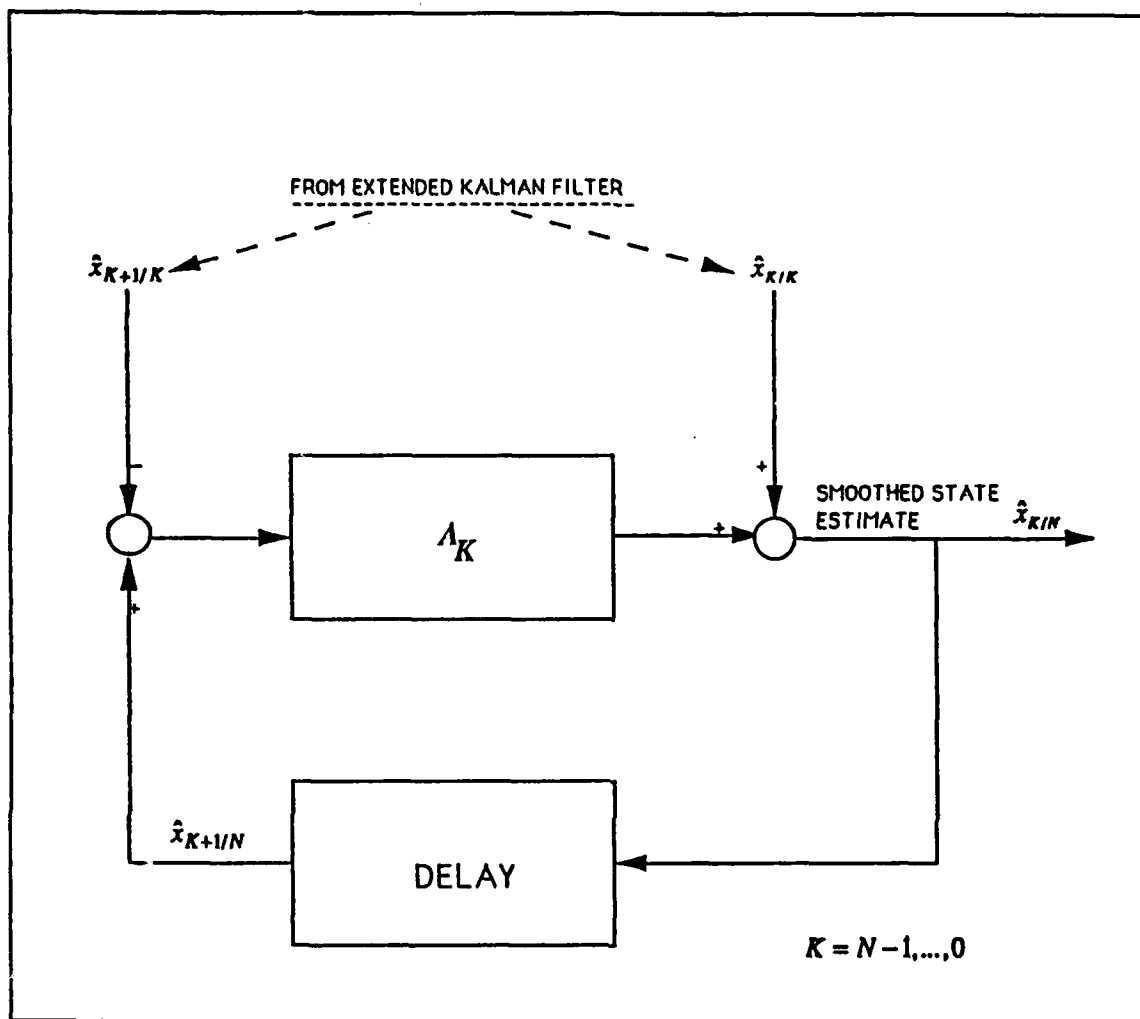
Figure 5. Block Diagram of the Smoothing Filter.

$$P_{K/N} = P_{K/K} + A_K[P_{K+1/N} - P_{K+1/K}]A_K^T \qquad (3.15)$$

where

A_K = smoothing gain matrix,

$\hat{x}_{K/N}$ = smoothed estimate at time K,

$P_{K/N}$ = smoothed error covariance at time K,

$\hat{x}_{K/K}$ and $P_{K/K}$ = state estimate and error covariance stored by the extended Kalman

16

filter routine and

$\hat{x}_{K+1/K}$ and $P_{K+1/K}$ = predicted state estimate and predicted error covariance stored

by the extended Kalman filter routine.

Several sources were helpful in understanding these equations. [Refs. 7,8,11] As it can be seen from equation (3.15), the smoothed estimate provided by a fixed-interval smoothing algorithm is simply the extended Kalman filter estimate adjusted by a weighted error term. The error term is the difference between the smoothed estimate calculated for the previous data point and the predicted estimate calculated by the extended Kalman filter. It is also clear that the fixed-interval smoothing algorithm uses the values of $\hat{x}_{K/K}$ and $\hat{x}_{K+1/K}$ which are stored in the Kalman filter routine for each iteration. Additionally the values of $P_{K/K}$ and $P_{K+1/K}$ must be provided for the smoothing routine.

IV. THE NOISE PROCESS IN FIXED-INTERVAL SMOOTHING ALGORITHM.

A. GENERAL

This work is devoted to studying the effects of the state excitation matrix Q_K in the smoothing algorithm. To accomplish this, the magnitude of this matrix is changed during the assumed maneuver periods and the effects of these changes on the smoothing algorithm's accuracy are investigated. Also, a new maneuver detection technique is developed to determine the maneuver periods.

B. NOISE PROCESS

The state excitation matrix Q_K represents the system noise process. This matrix is a function of the acceleration matrix a_K and the noise transition matrix Γ_K, so that

$$Q_K = \left[\hat{\omega}_K \hat{\omega}_K^T\right] \tag{4.1}$$

where $\hat{\omega}_K$ is given by equation (2.4). Substituting equation (2.4) into equation (4.1), we find

$$Q_K = \Gamma_K \begin{bmatrix} E\left[a_{x_K}^2\right] & E\left[a_{xy_K}\right] \\ E\left[a_{yx_K}\right] & E\left[a_{y_K}^2\right] \end{bmatrix} \Gamma_K^T \tag{4.2}$$

For reference, the noise transition matrix Γ_K is given by equation (2.5). The Q_K matrix allows for any random target maneuvers and also serves to account for any model inaccuracies. These inaccuracies are the differences between the true action of the target and its motion as characterized by equation (2.1). Q_K also prevents the gain matrix G_K from approaching zero by ensuring some uncertainty in the predicted state error

covariance matrix $P_{K+1/K}$. By substituting equation (2.5) into, the equation (4.2), equation (4.2) can be expanded as follows:

$$
Q_K = \begin{bmatrix}
\frac{1}{4} E[a_{x_K}^2] T^4 & \frac{1}{2} E[a_{x_K}^2] T^3 & \frac{1}{4} E[a_{xy_K}] T^4 & \frac{1}{2} E[a_{xy_K}] T^3 \\
\frac{1}{2} E[a_{x_K}^2] T^3 & E[a_{x_K}^2] t^2 & \frac{1}{2} E[a_{xy_K}] T^3 & E[a_{xy_K}] T^2 \\
\frac{1}{4} E[a_{yx_K}] T^4 & \frac{1}{2} E[a_{yx_K}] T^3 & \frac{1}{4} E[a_{y_K}^2] T^4 & \frac{1}{2} E[a_{y_K}^2] T^3 \\
\frac{1}{2} E[a_{yx_K}] T^3 & E[a_{yx_K}] T^2 & \frac{1}{2} E[a_{y_K}^2] T^3 & E[a_{y_K}^2] T^2
\end{bmatrix} \tag{4.3}
$$

The velocity of the target can be described in terms of its linear velocity and heading. From Figure 1 on page 6, this relationship is given as

$$v_x = v_t \sin \Theta_t \tag{4.4}$$

$$v_y = v_t \cos \Theta_t \tag{4.5}$$

By differentiating equations (4.3) and (4.4) we obtain the target's acceleration in the x and y directions:

$$a_x = \dot{v}_t \sin \Theta_t + v_t \dot{\Theta}_t \cos \Theta_t$$

$$a_x = \dot{v}_t \frac{v_x}{v_t} + v_t \dot{\Theta}_t \frac{v_y}{v_t}$$

$$a_x = \dot{v}_t \frac{v_x}{v_t} + \Theta_t v_y \tag{4.6}$$

and

$$a_y = \dot{v}_t \cos \Theta_t - v_t \dot{\Theta}_t \sin \Theta_t$$

19

$$a_y = \dot{v}_t \frac{v_y}{v_t} + v_t \dot{\Theta}_t \frac{v_x}{v_t}$$

$$a_y = \dot{v}_t \frac{v_y}{v_t} + \Theta_t v_x \tag{4.7}$$

The noise is initially described by

$$E[\dot{v}_t] = E[\dot{\Theta}_t] = 0 \tag{4.8}$$

$$E[\dot{v}_t] = \sigma_{v_t}^2 \tag{4.9}$$

and

$$E[\dot{\Theta}_t] = \sigma_{\Theta_t}^2 \tag{4.10}$$

By squaring equations (4.6) and (4.7) and taking the expectations, the variances of target's accelerations, a_x and a_y, are:

$$E[a_{x_K}^2] = \left[\frac{v_x}{v_t} \right]^2 \sigma_{v_t}^2 + v_y^2 \sigma_{\Theta_t}^2 \tag{4.11}$$

and

$$E[a_{y_K}^2] = \left[\frac{v_y}{v_t} \right]^2 \sigma_{v_t}^2 + v_x^2 \sigma_{\Theta_t}^2 \tag{4.12}$$

We also find that the covariance of a_x and a_y denoted by a_{xy} or a_{yx} is

$$E[a_{xy_K}] = E[a_{yx_K}] = v_x v_y \left[(\frac{\sigma_{v_t}^2}{v_t})^2 - \sigma_{\Theta_t}^2 \right] \tag{4.13}$$

20

From [Ref. 12], the position of the target assuming speed is constant

$$x_{K+1} = x_K + v_{x_K}T \tag{4.14}$$

$$y_{K+1} = y_K + v_{y_K}T \tag{4.15}$$

and the position of the target assuming acceleration is constant:

$$x_{K+1} = x_K + v_{x_K}T + \frac{1}{2}a_{x_K}T^2 \tag{4.16}$$

$$y_{K+1} = y_K + v_{y_K}T + \frac{1}{2}a_{y_K}T^2 \tag{4.17}$$

By comparing the equations (4.14) thorough (4.17), it can be seen that the expected position errors due to the unknown accelerations of the target can be defined as

$$E[\tilde{x}_{K+1}] = \frac{1}{2}E[a_{x_K}]T^2 \tag{4.18}$$

and

$$E[\tilde{y}_{K+1}] = \frac{1}{2}E[a_{y_K}]T^2 \tag{4.19}$$

The variances of these errors are

$$E[\tilde{x}_{K+1}^2] = \frac{1}{4}E[a_{x_K}^2]T^4 \tag{4.20}$$

and

$$E[\tilde{y}_{K+1}^2] = \frac{1}{4}E[a_{y_K}^2]T^4 \tag{4.21}$$

21

By comparing equations (4.20) and (4.21) with equation (4.3), we see that these equations are equal to elements (1,1) and (3,3) of the Q_K matrix. Out of all the elements in the Q_K matrix, these two elements have the greatest effect in compensating the position errors. Since it is most important to compensate the error on the axis which has the maximum error variance, the algorithm developed determines the Q_K matrix using these elements for the magnitude of the Q_K matrix. This algorithm first compares the error variances on the x and y axes, σ_x and σ_y, to determine which axis has the greater error variance. If $\sigma_x > \sigma_y$, Q_K matrix becomes

$$Q_K = Q_{(1,1)}I \tag{4.22}$$

where I is the unity matrix and

$$Q_{(1,1)} = \frac{1}{4} E\left[a_{x_K}^2\right]T^4$$

If $\sigma_x < \sigma_y$, the Q_K matrix is

$$Q_K = Q_{(3,3)}I \tag{4.23}$$

where

$$Q_{(3,3)} = \frac{1}{4} E\left[a_{y_K}^2\right]T^4$$

C. THE STATE EXCITATION MATRIX IN THE FIXED-INTERVAL SMOOTHING ALGORITHM

As mentioned before, the fixed-interval smoothing filter uses as input the state estimates and error covariances calculated by the forward-time Kalman filter. But in order to see the effects of the state excitation matrix Q_K in the smoothing algorithm, the predicted error covariance matrix $P_{K+1/K}$ is recalculated in the smoothing routine. The pre-

dicted state estimates $\hat{x}_{K+1/K}$ are also recalculated in the smoothing routine. By recalculating these matrices, we attempted to get a feeling for the expected magnitude of the smoothing error. The intent was to enable the the filter to carry along its own error analysis. The new system of the recursive equations for the fixed-interval smoothing becomes: [Refs. 4,13]

$$\hat{x}_{K+1/K} = \phi \hat{x}_{K/K} \tag{4.24}$$

$$P_{K+1/K} = \phi P_{K/K}\phi^T + Q_K \tag{4.25}$$

$$A_K = P_{K/K}\phi^T P_{K+1/K}^{-1} \tag{4.26}$$

$$\hat{x}_{K/N} = \hat{x}_{K/K} + A_K[\hat{x}_{K+1/N} - \hat{x}_{K+1/K}] \tag{4.27}$$

$$P_{K/N} = P_{K/K} + A_K[P_{K+1/N} - P_{K+1/K}]A_K^T \tag{4.28}$$

As seen from equation (4.26), the smoothing filter gains are a function of the error covariance. As the predicted error increases, the smoothing gains decrease due to the inverse relationship between smoothing gains and the predicted error covariance matrix. In this way the smoothing filter can compensate for a large expected error by placing more emphasis on the Kalman filter estimates. By substituting equation (4.24) into equation (4.27), we obtain

$$\hat{x}_{K/N} = \hat{x}_{K/K} + A_K[\hat{x}_{K+1/N} - \phi \hat{x}_{K/K}]$$

$$\hat{x}_{K/N} = [I - A_K\phi]\hat{x}_{K/K} + A_K\hat{x}_{K+1/N} \tag{4.28}$$

Equation (4.28) shows that a small A_K causes the factor of $[I - A_K\phi]$ to approach unity, thereby placing more emphasis on the forward-time Kalman filter estimates $\hat{x}_{K/K}$.

We can exploit this behavior of the smoothing filter and use it to adapt the smoothing filter to detected target maneuvers.

D. MANEUVER DETECTION

Should the target maneuver during the tracking process, the filtered estimates tend to diverge from the true estimates. This introduces error into the state estimates. Therefore a procedure must be developed to detect the target's maneuvers. This can be accomplished by monitoring the filter residual process.

The residual process of the extended Kalman filter is taken as the difference between the observed position and the filter's predicted position estimates. This process can be defined as

$$|z_K - \hat{x}_{K/K-1}| \tag{4.29}$$

The maneuver detection technique implemented calculates the residual value for each observation and compares this to the two maneuver gates. The gates are defined as three times and eight times the predicted standard deviation. Some of the principles underlying this technique are presented in [Ref. 14]. To define the predicted standard deviation, error ellipse equations are used. More detailed information about error ellipses can be found in [Ref. 6: pp. 17-18]. These equations are:

$$\sigma_{x'}^2 = \frac{\sigma_x^2 + \sigma_y^2}{2} + \frac{cov(xy)}{\sin 2\theta} \tag{4.30}$$

$$\sigma_{y'}^2 = \frac{\sigma_x^2 + \sigma_y^2}{2} - \frac{cov(xy)}{\sin 2\theta} \tag{4.31}$$

where

σ_x^2 and σ_y^2 = variances in the original cartesian coordinate system,

$\sigma_{x'}^2$ and $\sigma_{y'}^2$ = variances along the major and minor axis oriented by

$$\theta = \frac{1}{2} \tan \left[\frac{2cov(xy)}{\sigma_x^2 - \sigma_y^2} \right]$$ (4.32)

By taking the square roots of equations (4.30) and (4.31), the standard deviations on the x′ and y′ axes of the error ellipse are

$$\sigma_{x'} = \sqrt{\frac{\sigma_x^2 + \sigma_y^2}{2} + \frac{cov(xy)}{\sin 2\theta}}$$ (4.33)

and

$$\sigma_{y'} = \sqrt{\frac{\sigma_x^2 + \sigma_y^2}{2} - \frac{cov(xy)}{\sin 2\theta}}$$ (4.34)

The maneuver detection algorithm compares the two standard deviations which are represented by the lengths of the x′ and y′ axes of the error ellipse shown in Figure 6. It selects the larger one as a predicted standard deviation, allowing the gates to take on the following values:

$$LOWER\_GATE = 3\sigma_K$$

and

$$UPPER\_GATE = 8\sigma_K$$

where σ_K is the larger of the two standard deviations, $\sigma_{x'}$ or $\sigma_{y'}$.

The reason for choosing the value of $3\sigma_K$ for the lower gate is well explained in [Ref. 14]. The value of the upper gate, known as a "Glitch" gate, is dependent on the operational characteristics of the target. This gate rejects motions that the target could not possibly make. In our problem, extremely high linear or tangential accelerations are examples of such behavior. When the residual exceeds this gate, the filter recognizes

25
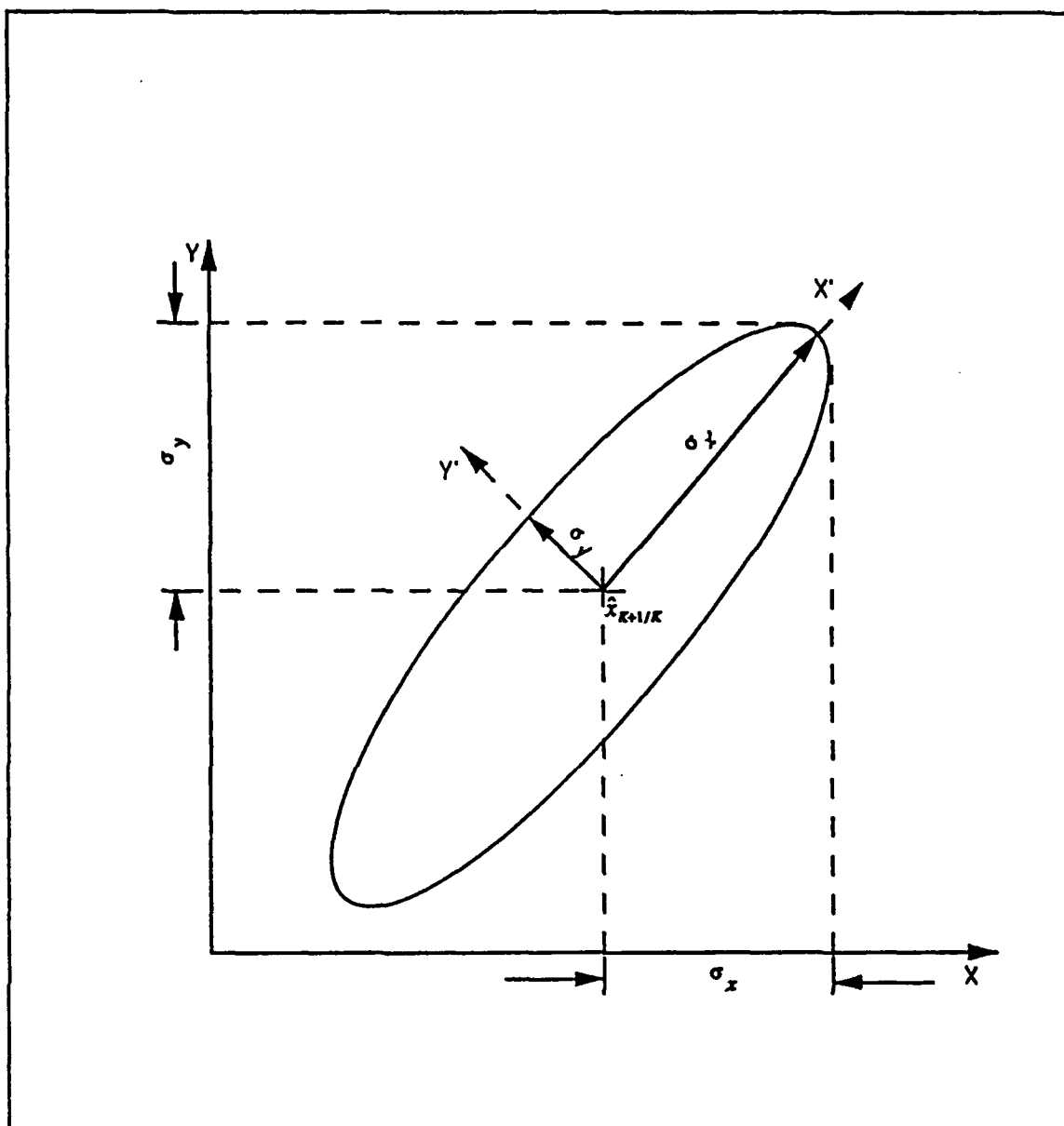
Figure 6.   Error Ellipse

that this motion is impossible for the given target and so must be due to noise. The value of $8\sigma_R$ gave the best results in this application.

For each observation, the calculated residual is compared to the two gates by the maneuver detection algorithm in the extended Kalman filter routine. If the residual is

less than the value of the lower gate, the filter continues on and processes the next observation. If the residual is larger than the value of the upper gate, the filter ignores that observation by setting filter gains equal to zero, thereby making the state estimates equal to the predicted estimates,

$$\hat{x}_{K+1/K+1} = \hat{x}_{K+1/K} \qquad (4.35)$$

This procedure will work well for isolated bad observations. However, if there are several consecutive bad observations, the filter can conceivably lose track of the target as the filter's state estimates diverge more and more away from the actual target states. To remedy this, the extended Kalman filter sets the filter gains equal to zero only for the first of two consecutive bad observations but uses non zero gain for the second. If the residuals of the two consecutive observations are in the zone between the two maneuver gates, shown as concentric circles in Figure 7, a maneuver is detected and compensation algorithm begins. The value of two provides a trade-off between fast response and low false alarm rates.

The maneuver detection algorithm does not run a second time in the fixed-interval smoothing routine, since it can use the maneuver times detected in the extended Kalman filter with no loss of accuracy. Additionally, the fixed-interval smoothing algorithm "backs up" and considers the first point ignored by the Kalman filter as a maneuver point, since it knows that if the maneuver is detected at some observation time in the Kalman filter routine, it must have started one observation earlier.

During the compensation, the state excitation matrix Q_K is increased by multiplying the coefficients along the main diagonal by a factor of 2.0. These coefficients account for random course and speed changes of the target. As the Q_K matrix is increased, the predicted error covariance $P_{K+1/K}$ is also increased because of the direct effect of the Q_K matrix on th magnitude of the predicted error covariance. And, as the $P_{K+1/K}$ matrix
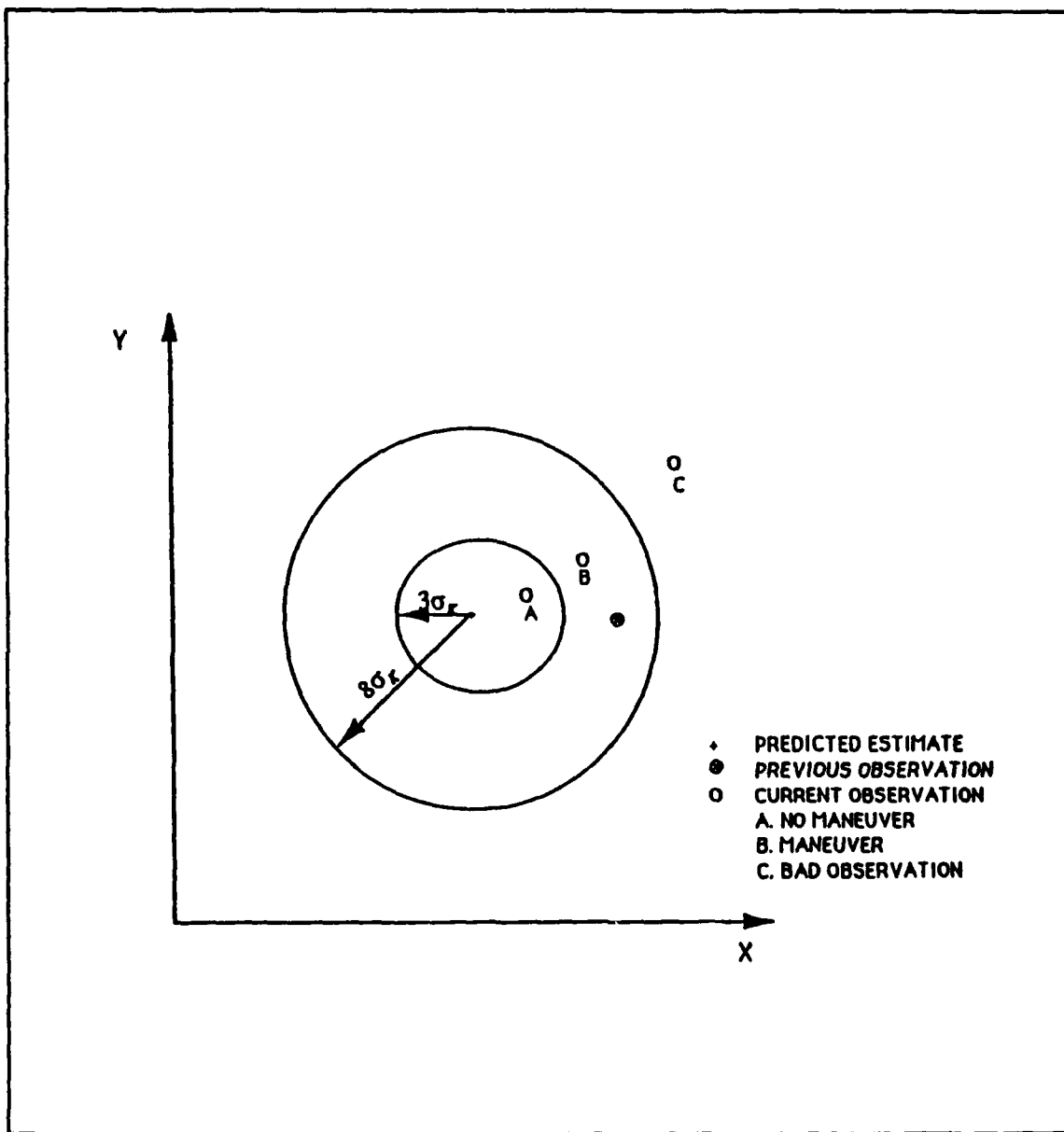
Figure 7. Diagramming the Maneuver Detection Technique

increases, the outputs of both the extended Kalman filter and the smoothing filter are affected as explained before. The multiplicative constant of 2.0 was found by trial and error.

V. COMPUTER SIMULATIONS

A. GENERAL

The SHIPTRACK.FOR extended Kalman filter algorithm was first implemented in [Ref. 3] on an Apple Macintosh Plus microcomputer. In [Ref. 4] the fixed-interval smoothing algorithm was added, the new algorithm was named SHIPSM.FOR this algorithm was adapted to run on an IBM PC. This research takes the program one step further by adding new algorithms to detect maneuvers and to derive the state excitation matrix Q_x. A new program SHIPMANE is used in the following manner for the simulations.

The raw data required by the SHIPMANE.FOR is generated by RAWDATA.FOR. This program is modified from the program TRACK.FOR used in [Ref. 3]. Our intention was to make the target follow a circular track during the maneuver period rather than make a sharp turn. Program RAWDATA.FOR asks the user for the initial positions, speeds and courses of the target and the tracking ships, the total tracking time and the observation interval. It also requests the desired maneuver period and any speed and course changes of the target during this period. The outputs consist of noisy or noise free bearings from each tracking ship to the target, the updated positions of all the vessels and the time of the observation are stored in the file called TRKDATA.DAT.

The program SHIPMANE.FOR reads and processes the data stored in TRKDATA.DAT. The outputs of this program are mainly stored in three files. The first file FILDATA.DAT stores the results of the extended Kalman filter portion of the program SHIPMANE.FOR while the fixed-interval smoothing results are included in the second file SMDATA.DAT. The results of the maneuver detection algorithm are stored in the third file MANEUDATA.DAT during the process of the extended Kalman

29

filter portion of the SHIPMANE.FOR. Additionally, a fourth file TRUDATA .DAT, is created for graphic purposes, and consists of the actual positions (tracks) of the target. Although this file is useful for the purposes of this thesis, in real world tracking problem this information would seldom, if ever, be available. In this thesis the terms "real" or "actual", when applied to tracks or maneuvers, refer to the data contained in this file. "Assumed" tracks or maneuvers refer to what is detected by the extended Kalman filter or the smoothing routine.

The MATLAB graphic routines are used to obtain the graphical representations of the data included in the output files of the SHIPMANE.FOR. Five graphic outputs are obtained for each simulation case except for the third case, which has only two. For all cases except the third, the first graph is a geographic plot which show extended Kalman filtered track versus the the actual and observed target tracks. The second graph compares the track resulting from the fixed-interval smoothing with the actual and observed target tracks. The third graph is the time plot showing the filtered, smoothed and observed position errors. The fourth is also a time plot and shows the residuals for each observation along with the threshold values of the upper and lower maneuver detection gates. In the third simulation case, only this graph is included. The fifth graph gives the overall results for each case. Due to the limited number of variables which can be used in the single MATLAB graphic package, the true track of the target was shown as a line without each observation point being shown.

Although both of the programs SHIPMANE.FOR and RAWDATA.FOR can easily be modified for multi-bearing measurements, the simulation cases used only two bearings per observation as measurements, one from each tracking ship. The set of the simulations studied in this chapter consists of following cases:

- Case #1: 60° maneuver toward tracking ships, with noiseless measurements.
- Case #2: 60° maneuver away from tracking ships, with noiseless measurements.

30

- Case #3: Test for the maneuver detection algorithm. (1) Case #1 with different maneuver period. (2) Case #2 with different maneuver period.
- Case #4: 60° maneuver toward tracking ships, with noisy measurements.
- Case #5: 120° maneuver toward tracking ships, with noisy measurements.
- Case #6: 60° maneuver away from tracking ships, with noisy measurements.
- Case #7: 120° maneuver away from tracking ships, with noisy measurements.

In all cases, the target ship starts at the position (-75,150). The initial course of the target is 090° and the initial speed of the target is 15 knots for each case. The speed of the target is held constant throughout the simulation cases. The initial positions of the tracking ships are (-40,0) and (-60,0), and courses and speeds are 030° and 10 knots for each case. The speeds and courses of the tracking ships are also held constant. The observation period is 30 minutes and all cases run for 450 minutes.

The success of the algorithm can be expressed by the percentage improvement between the total error in observed positions and the total errors in the filtered estimates and smoothed estimates, respectively. This percentage indicates of how much the extended Kalman filtering and the fixed-interval smoothing improve the position accuracy over the observations. For the extended Kalman filter, this percentage is simply the ratio between the the total error in the observed positions and the total error in the filtered estimates throughout the simulation case or time period of interest. In some cases this was recalculated specifically for maneuver periods. The percentage improvement due to the smoothing was similarly the ratio between the total error in the observed position estimates and the total error in the smoothed position estimates. Also, the average position errors due to the extended Kalman filter and the fixed-interval smoothing algorithm are given for the different cases. The average position error due to the extended Kalman filter is calculated by summing the position errors of the filtered position estimates and then dividing by the total number of observations. The average position error due to the smoothing routine is also found by summing the position errors of smoothed

estimates over the entire simulation (or in some cases, over the number of observations of interest) and dividing by the number of observations. The average position errors show how well the extended Kalman filter and fixed-interval smoothing algorithm work for a particular simulation case.

B. CASE #1

The target is steaming due east at 15 knots at the beginning of this case. Between time equals 150 minutes and and time equals 300 minutes, it makes a 60° course change toward the advancing tracking ships on a circular track. The results for the filtering and smoothing are shown in Figure 8 and Figure 9. Since there is no noise in the measurements, the observed track equals the true track. Although the measurements are not noisy, the filter estimates diverge very slightly for the first several observations. This initial error, shown in Figure 10, is due to the inaccuracy of the initial state estimates. This inaccuracy also causes the high values for the upper and lower maneuver gates for the first few observations, as can be seen in Figure 11. When the target starts its turn at time equals 150, the tracking error begins to increase. It decreases, however, as the filter regains the target track and it reaches zero one observation after the target finishes its maneuver.

The fixed-interval smoothing algorithm improves the position accuracy over the extended Kalman filter by an average of 35% during the real maneuver period, between time equals 150 and equals 300 minutes, and 22% during the overall simulation.

As can be seen in Figure 11, the residuals appear between the upper and lower maneuver gates for time equals 240, 270 and 300. Since two consecutive residuals between the upper and lower gate values are necessary for the maneuver to be detected, the extended Kalman filter recognizes times 270 and 300 as a maneuver period. Time 240 is ignored by the Kalman filter. The smoothing algorithm, however, does not ignore time 240, since it knows that if the maneuver was detected at time 270, it must have begun at time 240. Therefore, the maneuver period for the fixed-interval smoothing algorithm is taken as times 240, 270 and 300. The overall results of this case can be seen in Figure 12.

33

Figure 8. The Results of the Kalman Filter Tracking for Case #1

34
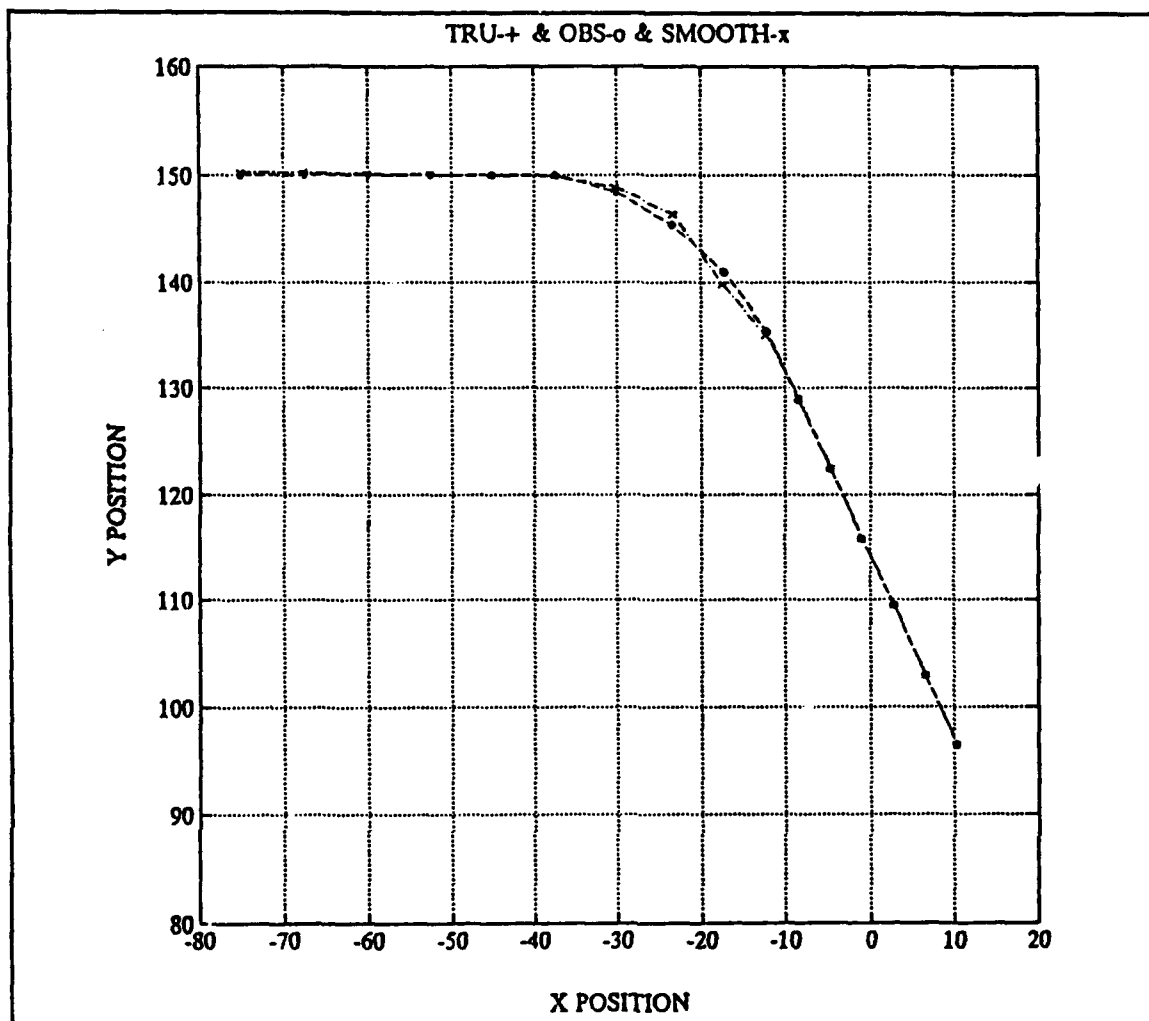
Figure 9. The Results of the Fixed-Interval Smoothing for Case #1

Figure 10. The Position Errors for Case #1
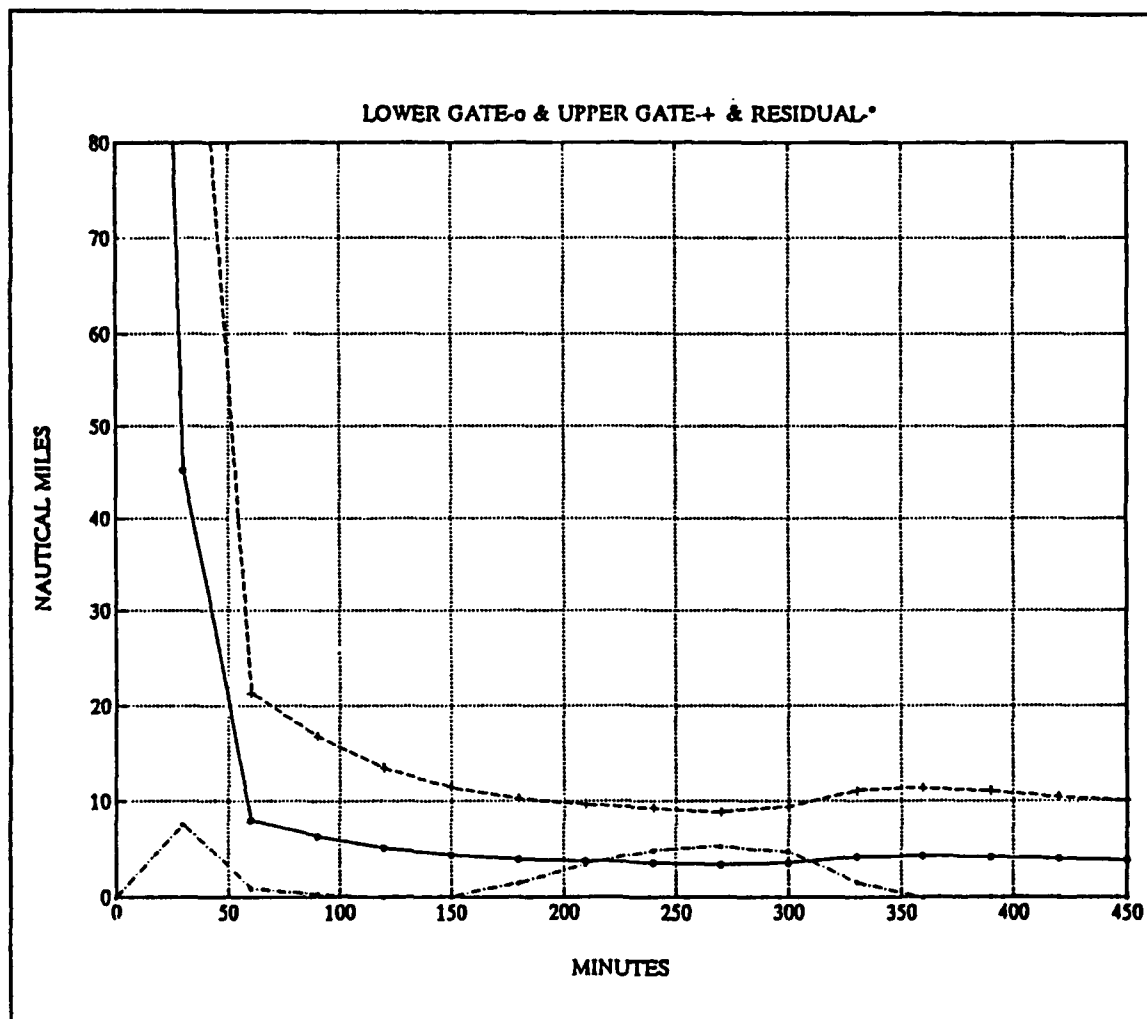
Figure 11.   The Results of the Maneuver Detection Algorithm for Case #1

37

TRU-. & OBS-o & FILTER-* & SMOOTH-x
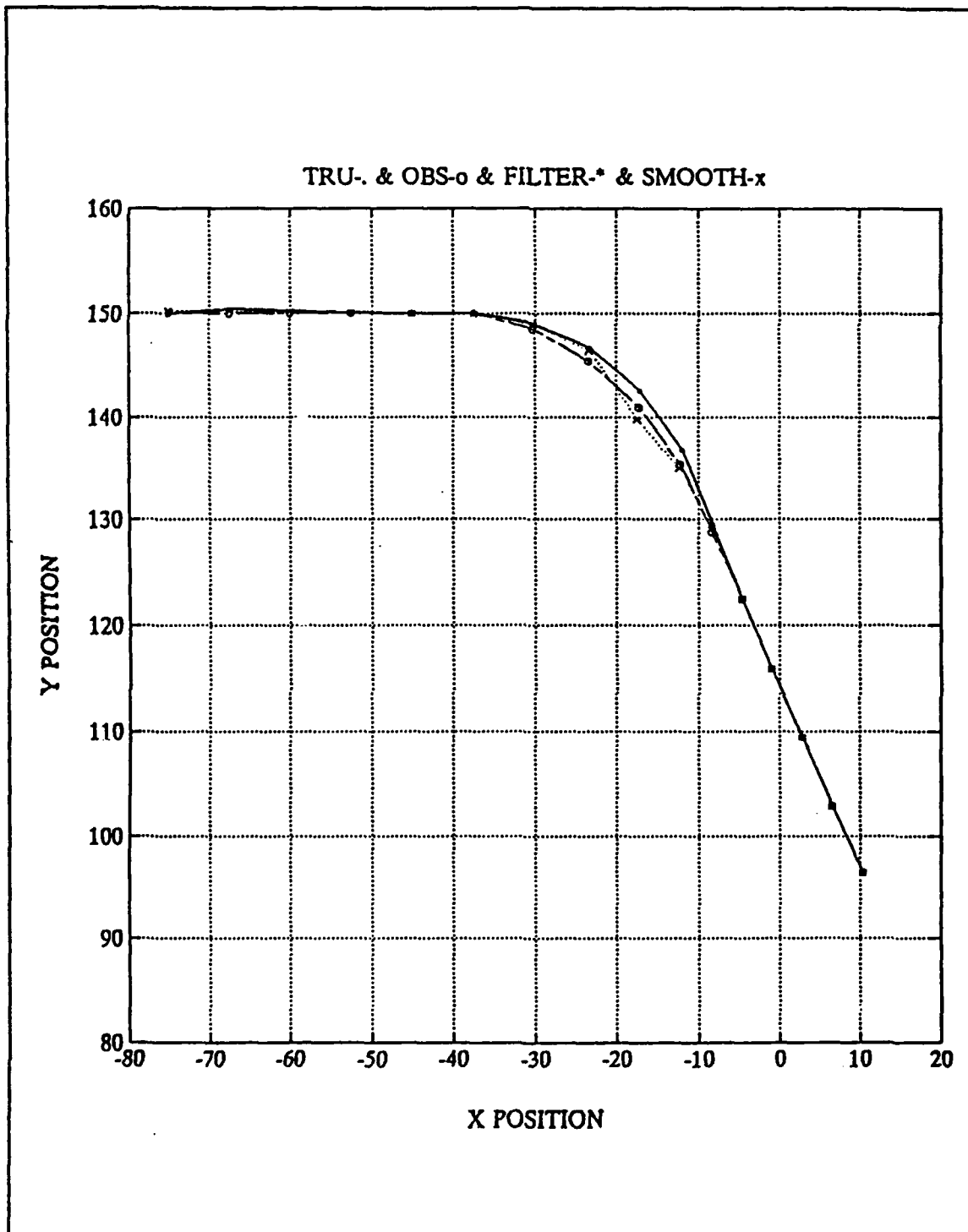
Figure 12. The Overall Results for Case #1

C.   CASE #2

In this case, the target makes a 60° maneuver away from the two tracking ships. The target's initial course is 090° at 15 knots. Between times 150 and 300, it turns northeast to a new course , 030°, on a circular track. Again, the observed and true tracks are the same due to the lack of measurement noise. T results of filtering are in Figure 13 and the results of the smoothing routine are in Figure 14. The initial error in Figure 15 and the high maneuver gate values for the first few observations in Figure 16 are again due to the error in the initial estimates.

The filter error starts to increase when the target begins to maneuver at time 150. When the target completes its maneuver, the error approaches zero. Since the target starts to pull away from the tracking ships, the filter error reaches zero later than it did in the previous case. From Figure 15, we can see how the fixed-interval smoothing routine improves the filter's estimate. The smoothing algorithm decreases the position error of the extended Kalman filter by an average of 22% for the overall case and by an average of 38% for the real maneuver period between 150 and 300 minutes.

From Figure 16, the residuals at times 240, 270 and 300 are in the maneuver zone. The maneuver period is detected for times 270 and 300 for the extended Kalman filter and for times 240, 270 and 300 for the smoothing filter. The final results of this case are in Figure 17.

Figure 13. The Results of the Kalman Filter Tracking for Case #2
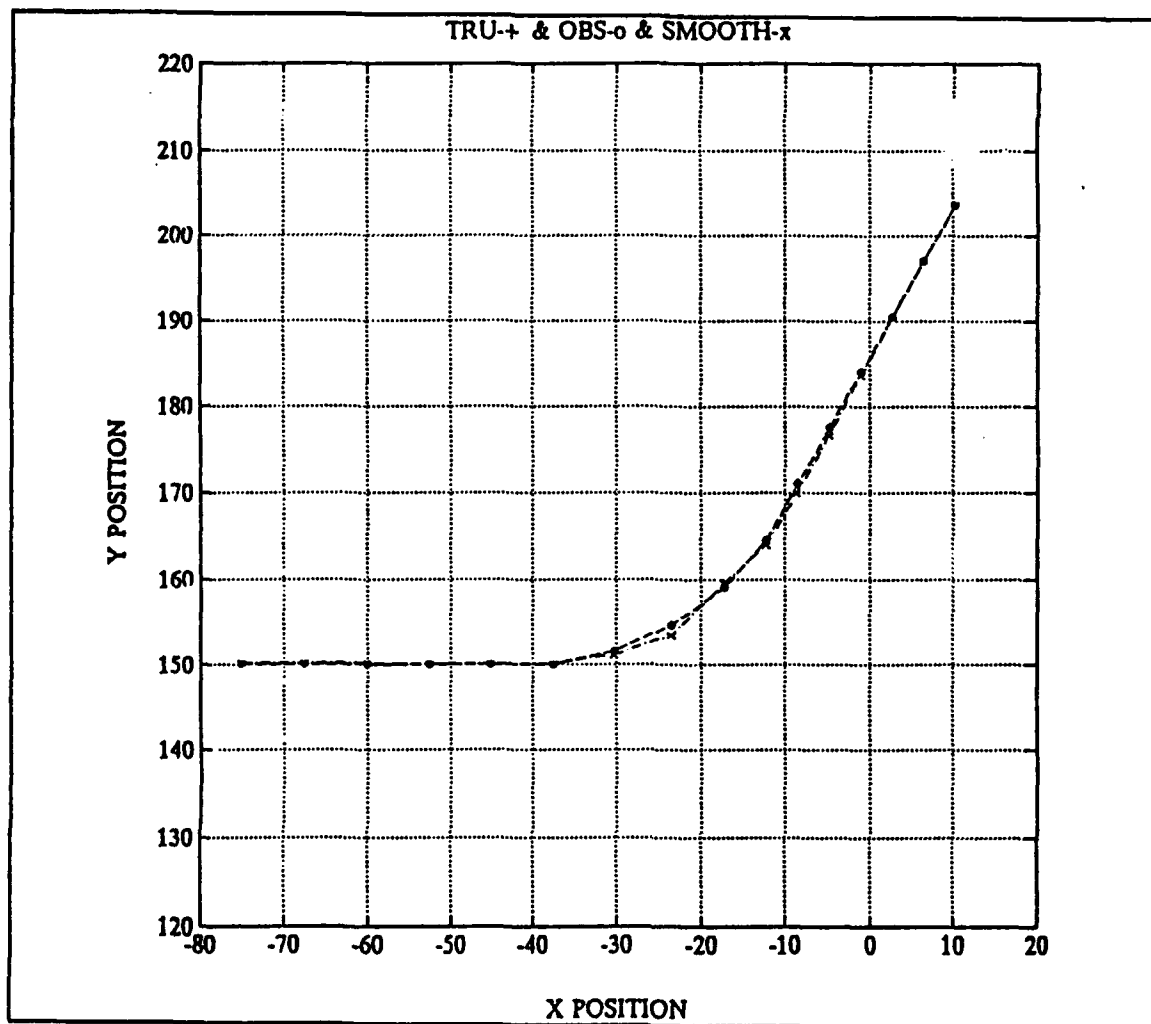
Figure 14. The Results of the Fixed-Interval Smoothing for Case #2

Figure 15. The Position Errors for Case #2

Figure 16. The Results of the Maneuver Detection Algorithm for Case #1

43

Figure 17. The Overall Results for Case #2

44

D. CASE #3

In this case, the two previous cases are tried with different maneuver periods in order to test the performance of the maneuver detection algorithm. Therefore only the figures which show the results of the maneuver detection algorithm are included.

1. Case #1 With Different Maneuver Period

In this part of the case, Case #1 is again tried with the new maneuver period from time 270 to time 390. From Figure 18, it can be seen that the residuals at times 300, 330, 360 and 390 are between the upper and lower gates. The maneuver period is between 330 and 390 minutes for the extended Kalman filter algorithm and between times 300 and 390 for the fixed-interval smoothing algorithm.

2. Case #2 With Different Maneuver Period

In this part, Case #2 with a new maneuver period, this time between 90 and 270 minutes, is simulated. In Figure 19, the residuals are in the maneuver detection zone at 180, 210, 240 and 270 minutes. In the extended Kalman filter routine, the maneuver detection algorithm detects the maneuver at 210, 240 and 270 minutes. For the fixed-interval smoothing algorithm, the maneuver period begins at time 180 and ends after time 270.
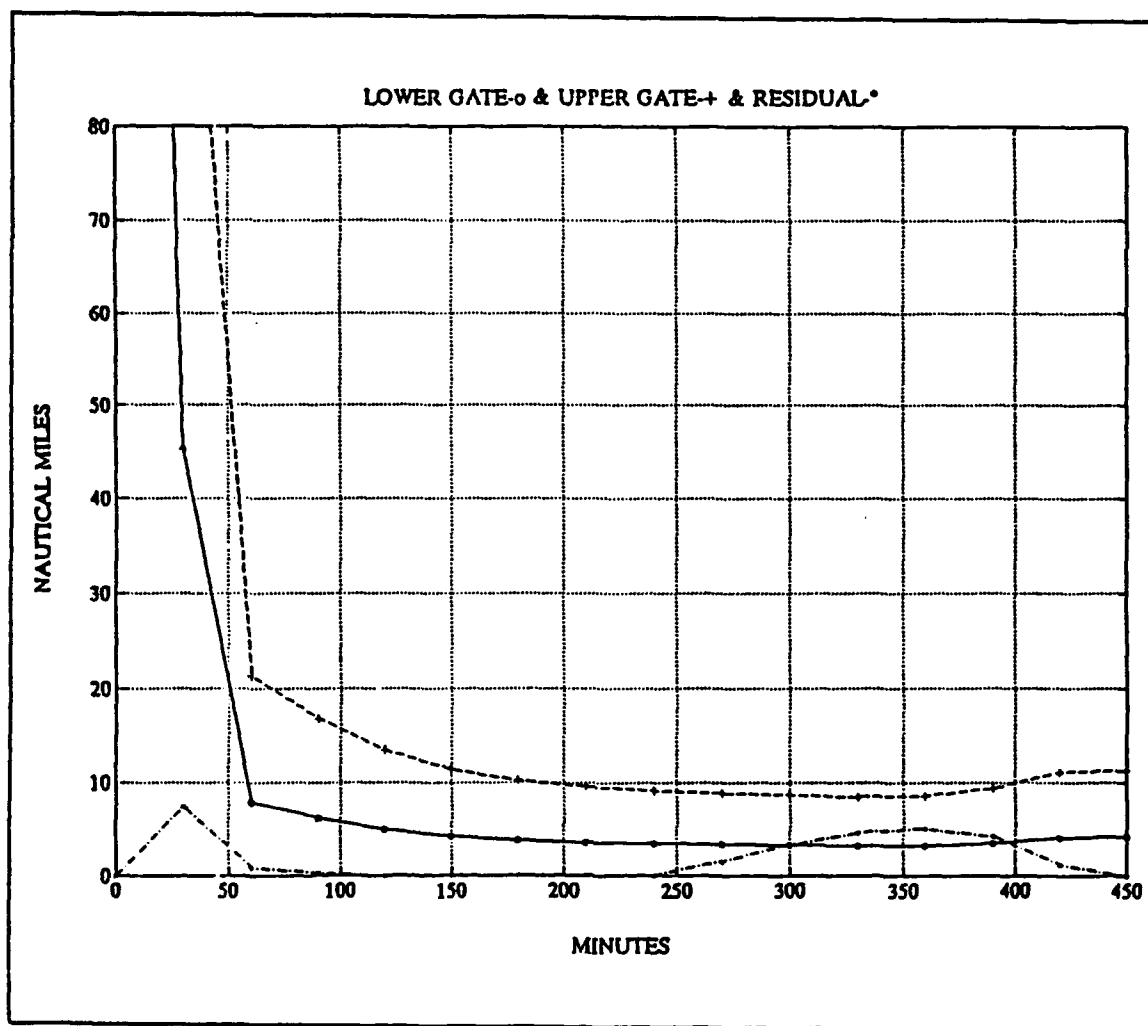
Figure 18. The Results of the Maneuver Detection Algorithm for Case #3-(1)
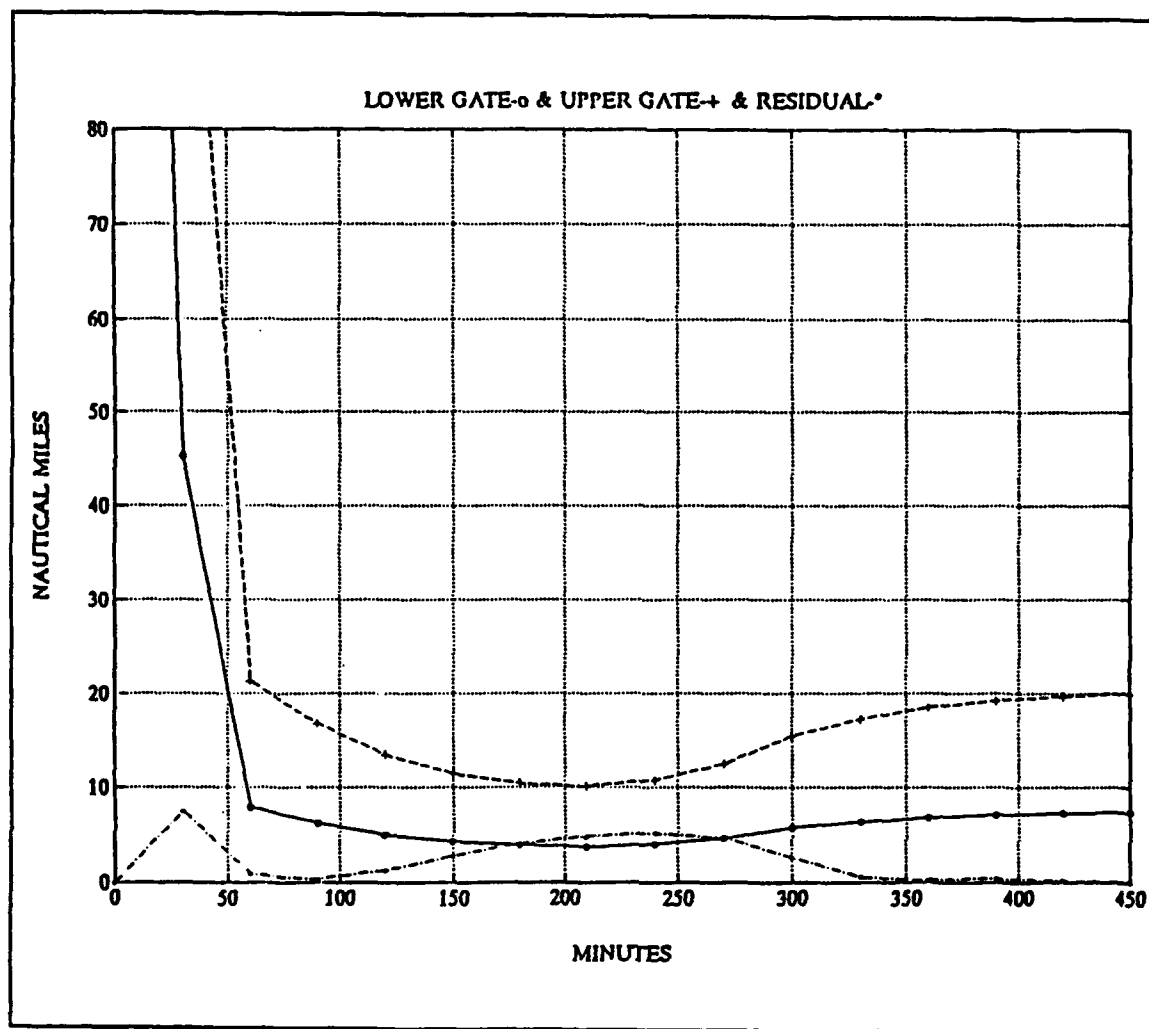
Figure 19. The Results of the Maneuver Detection Algorithm for Case #3-(2)

E. CASE #4

This case is the same as Case #1 except noise is added to the measurements. The filtering and smoothing results for this case are shown in Figure 20 and Figure 21. At the beginning of the tracking problem, the error is high. However, after the target maneuvers, the vessels close each other and the position error decreases rapidly.

From Figure 23, it is seen that the residuals are between the maneuver gates at times 180 through 330. The maneuver period for the extended Kalman filter is from 210 to 330 minutes. During this period the improvement in position error due to the Kalman filter is 32%. The maneuver period for the smoothing filter is between times 180 and 330, and the position error improvement due to the smoothing filter is 78%.

It is also seen that the maneuver detection algorithm recognizes a bad observation at time 60. As Figure 20 shows, the extended Kalman filter estimates for this point appear to be only the projections of the previous estimates in time. Therefore the position error of the extended Kalman filter for this point is 145% worse than the observed position error, while the smoothed position error is only 4% worse. The average position error is 3.8 Nm for the extended Kalman filter and 2.5 Nm for the smoothing filter over the entire tracking period. The overall results for this case are in Figure 24.
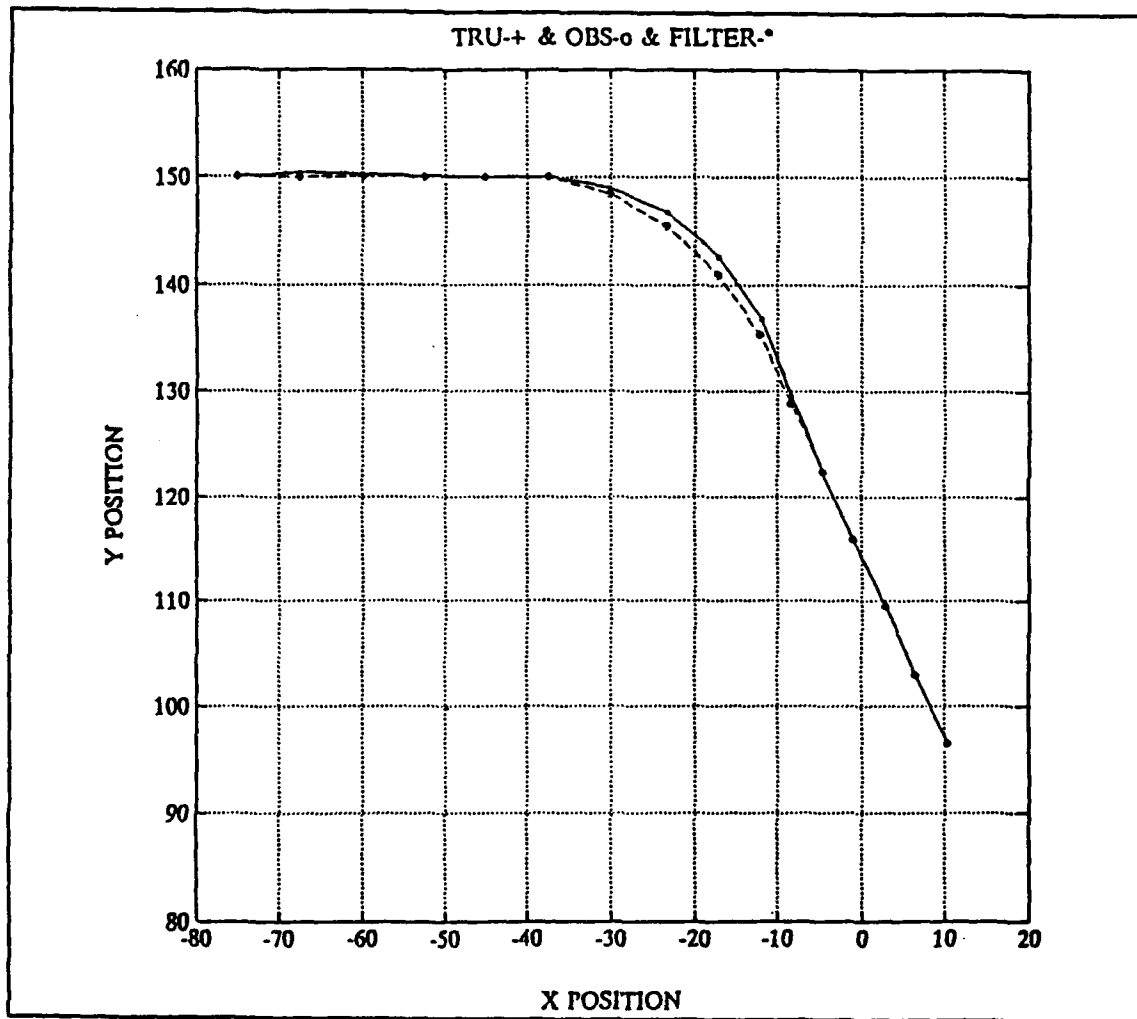
Figure 20.   The Results of the Kalman Filter Tracking for Case #4

Figure 21. The Results of the Fixed-Interval Smoothing for Case #4

Figure 22. The Position Errors for Case #4

Figure 23. The Results of the Maneuver Detection Algorithm for Case #4

TRU-. & OBS-o & FILTER-* & SMOOTH-x
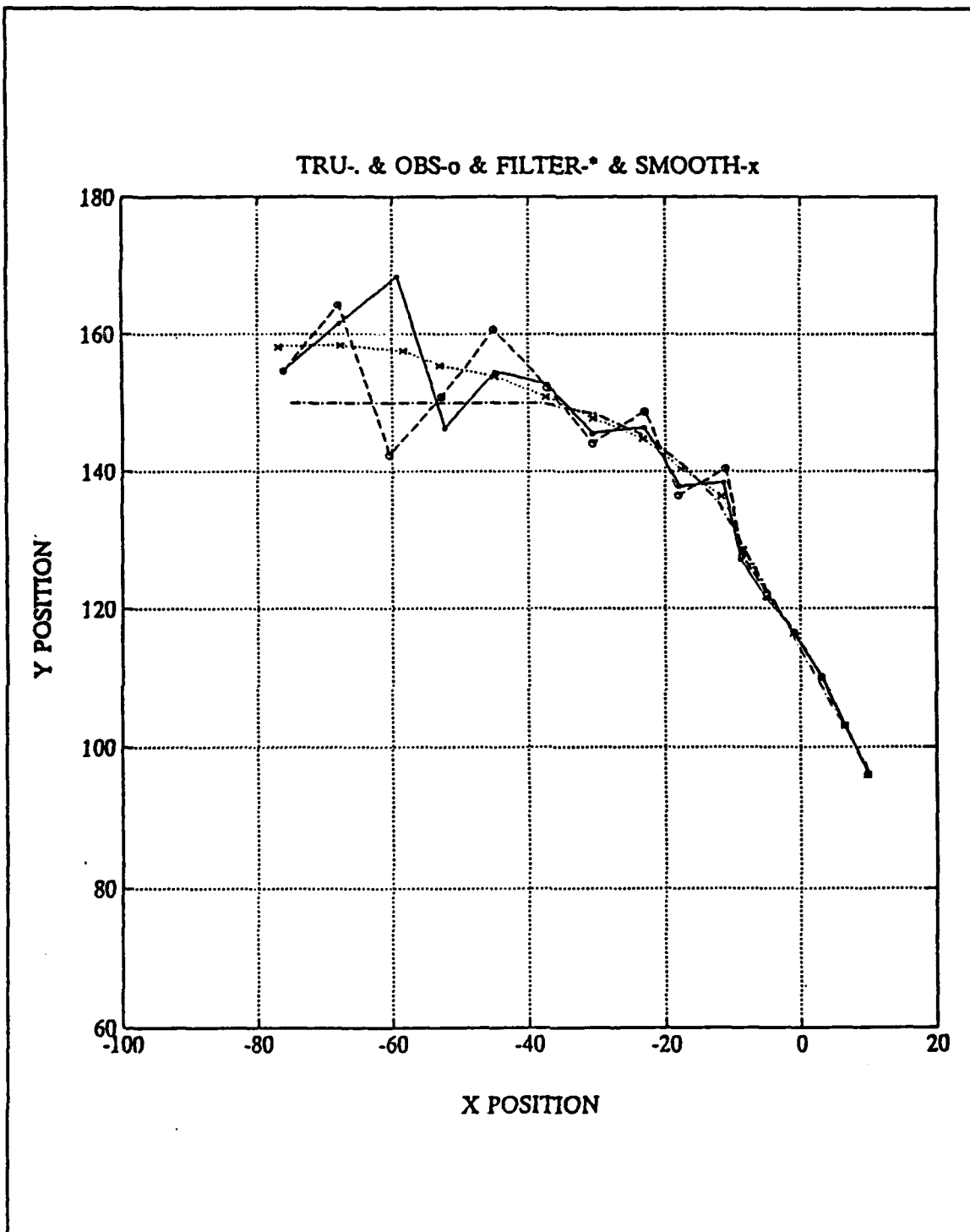
Figure 24. The Overall Results for Case #4

F. CASE #5

This case includes a 120° turn which, if undetected, will cause an unacceptably high error. The target turns to a new course of 210° by following a circular track between times 150 and 300. Again, the observed position error decreases rapidly after the target completes its maneuver, since all the vessels start to close each other. The extended Kalman-filtered and smoothed results can be seen in Figure 25 and Figure 26.

The observed, filtered and smoothed position errors are shown in Figure 27. The extended Kalman filter improves the position accuracy by 46% over the observed position errors for the entire tracking period, and the improvement due to smoothing is 73% over the same period. The maximum filtered position error is around 6 Nm except at time zero, while the average filtered error is 3 Nm. The maximum smoothed position error is 4 Nm and the average smoothed position error is 1.5 Nm.

The maneuver period for the extended Kalman filter is from 180 through 330 minutes, during which the improvement due to the extended Kalman filter is 46%. The maneuver period for the fixed-interval smoothing is between times 150 and 330, and the position accuracy is 55% over the observed position errors for the maneuver period alone. The observation at time equals 120 is recognized as a bad observation. The overall results are in Figure 29.
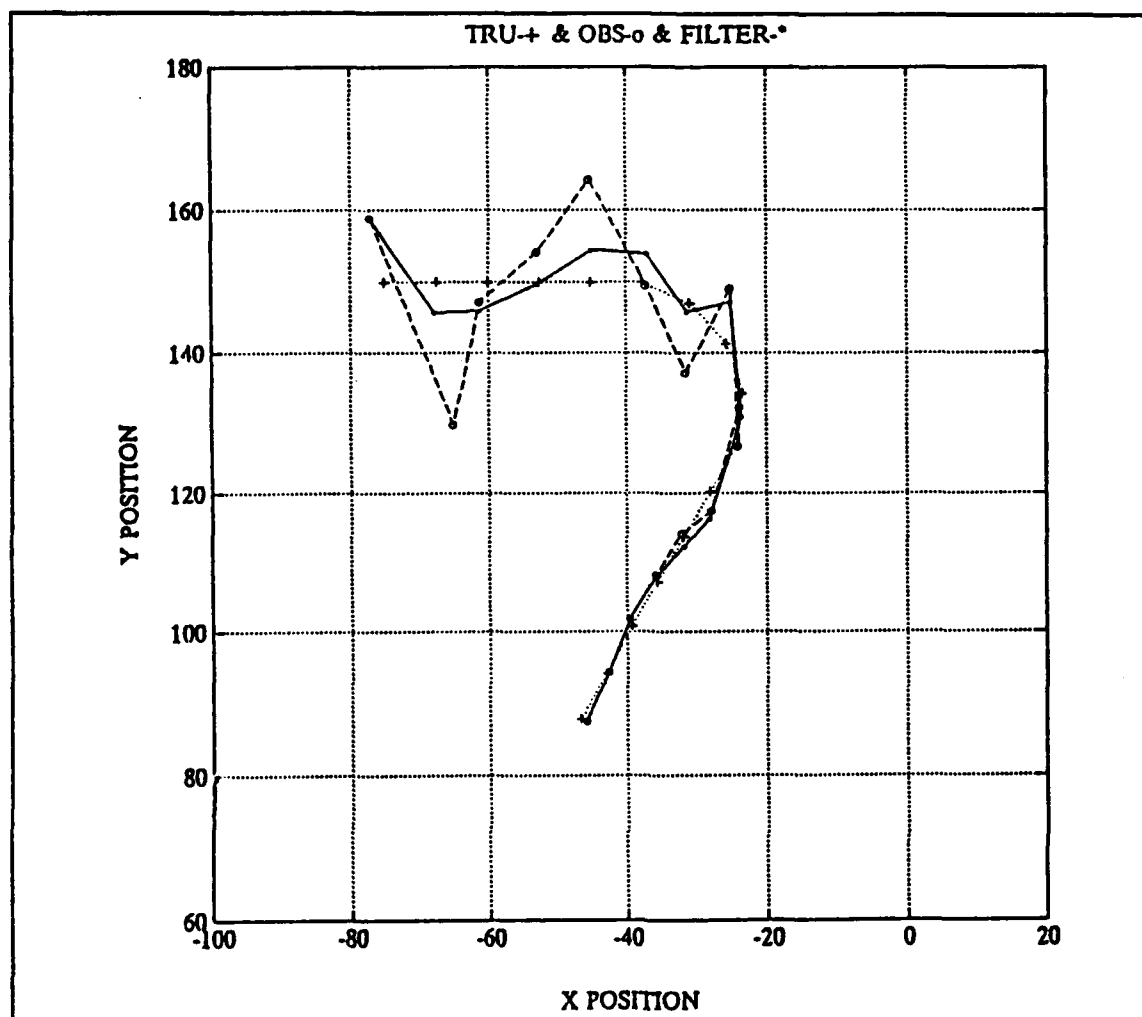
Figure 25. The Results of the Kalman Filter Tracking for Case #5
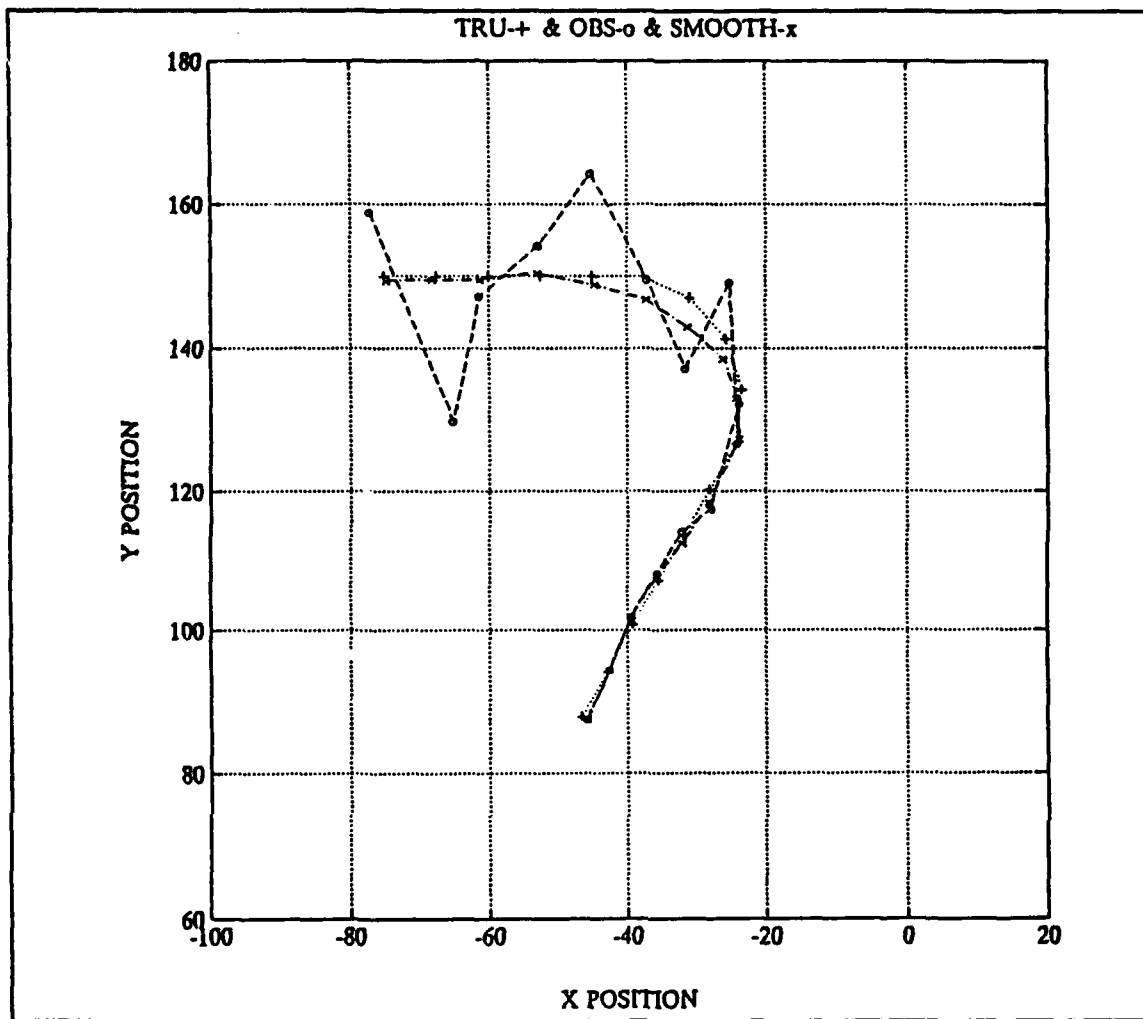
Figure 26. The Results of the Fixed-Interval Smoothing for Case #5

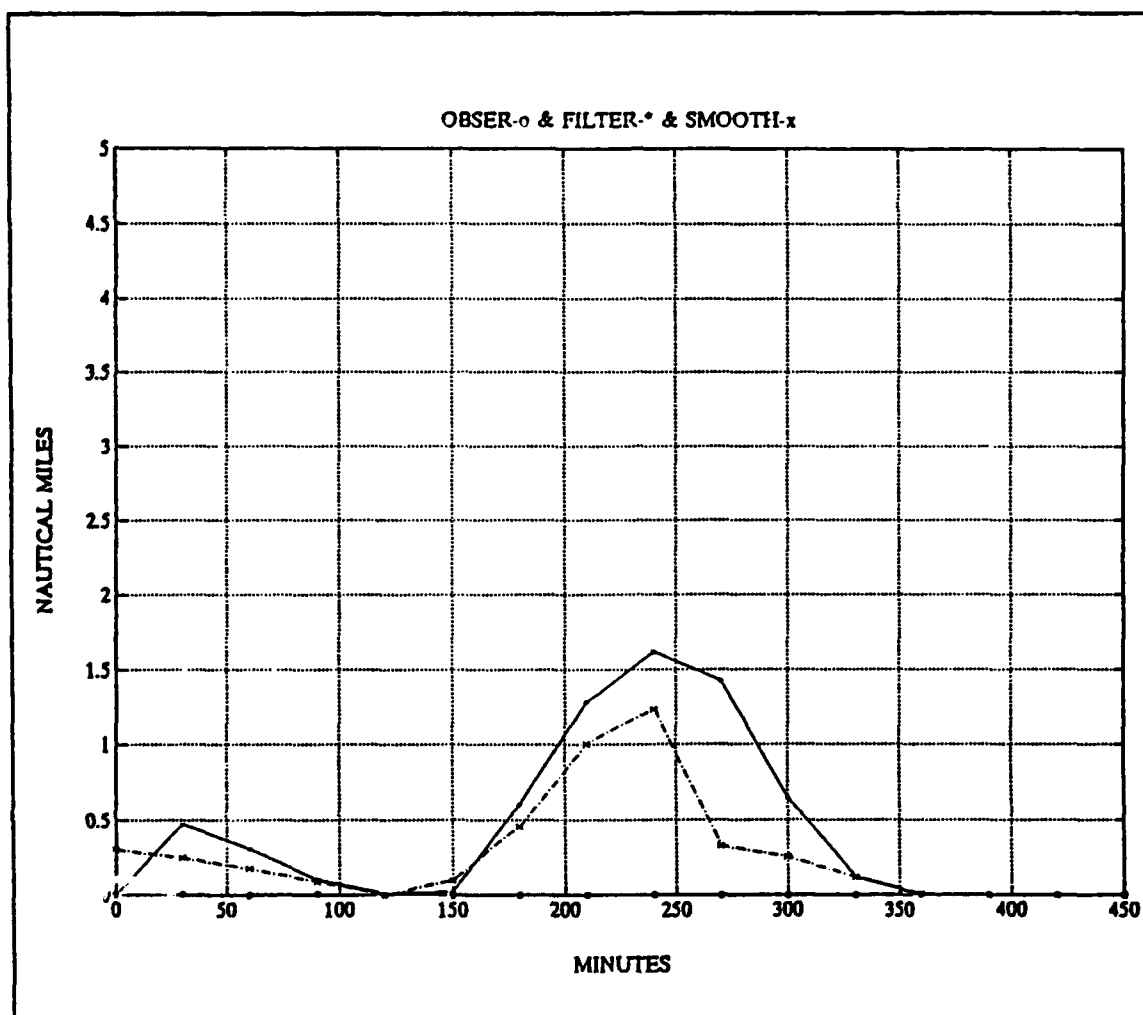**OBSER-o & FILTER-• & SMOOTH-x**
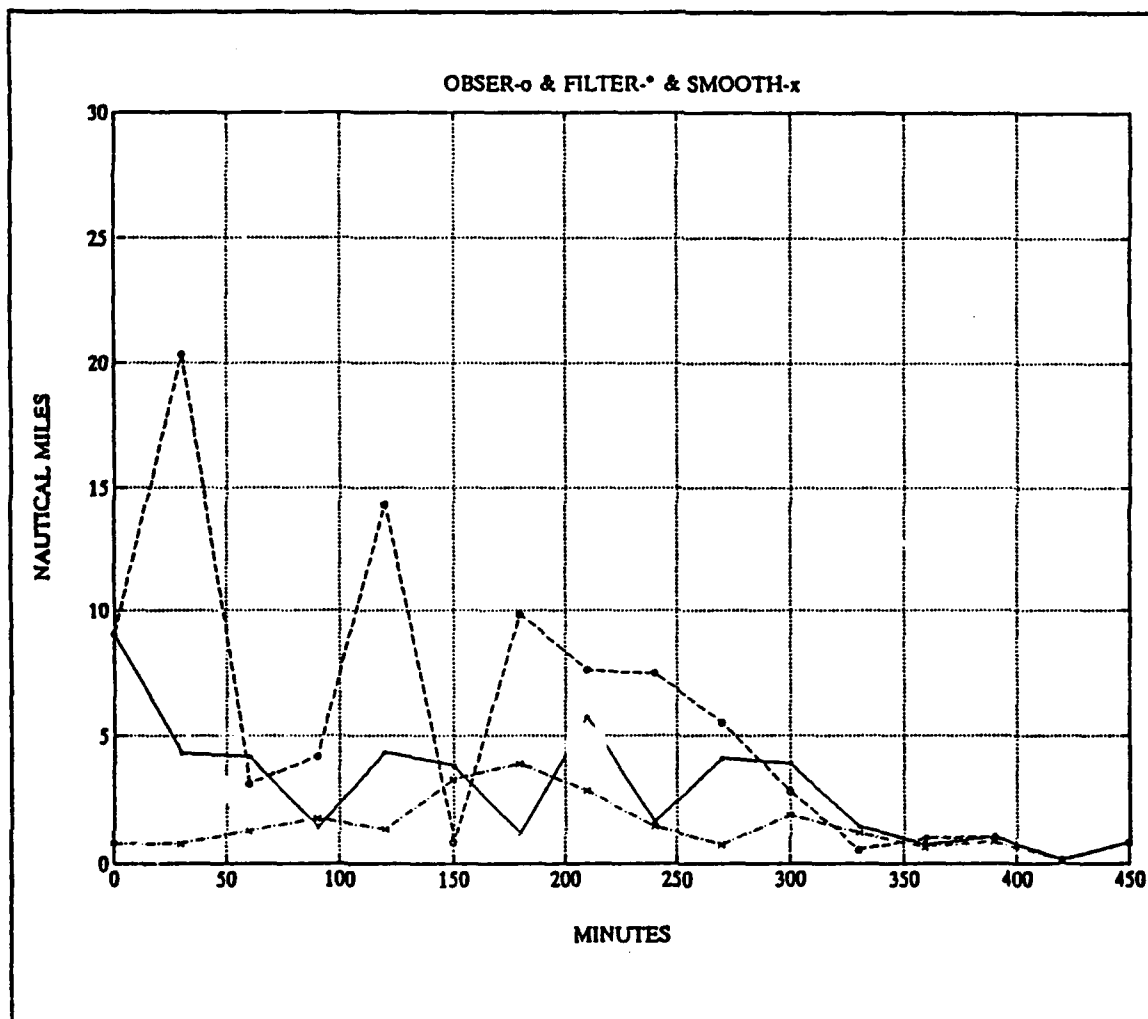
Figure 27.   The Position Errors for Case #5
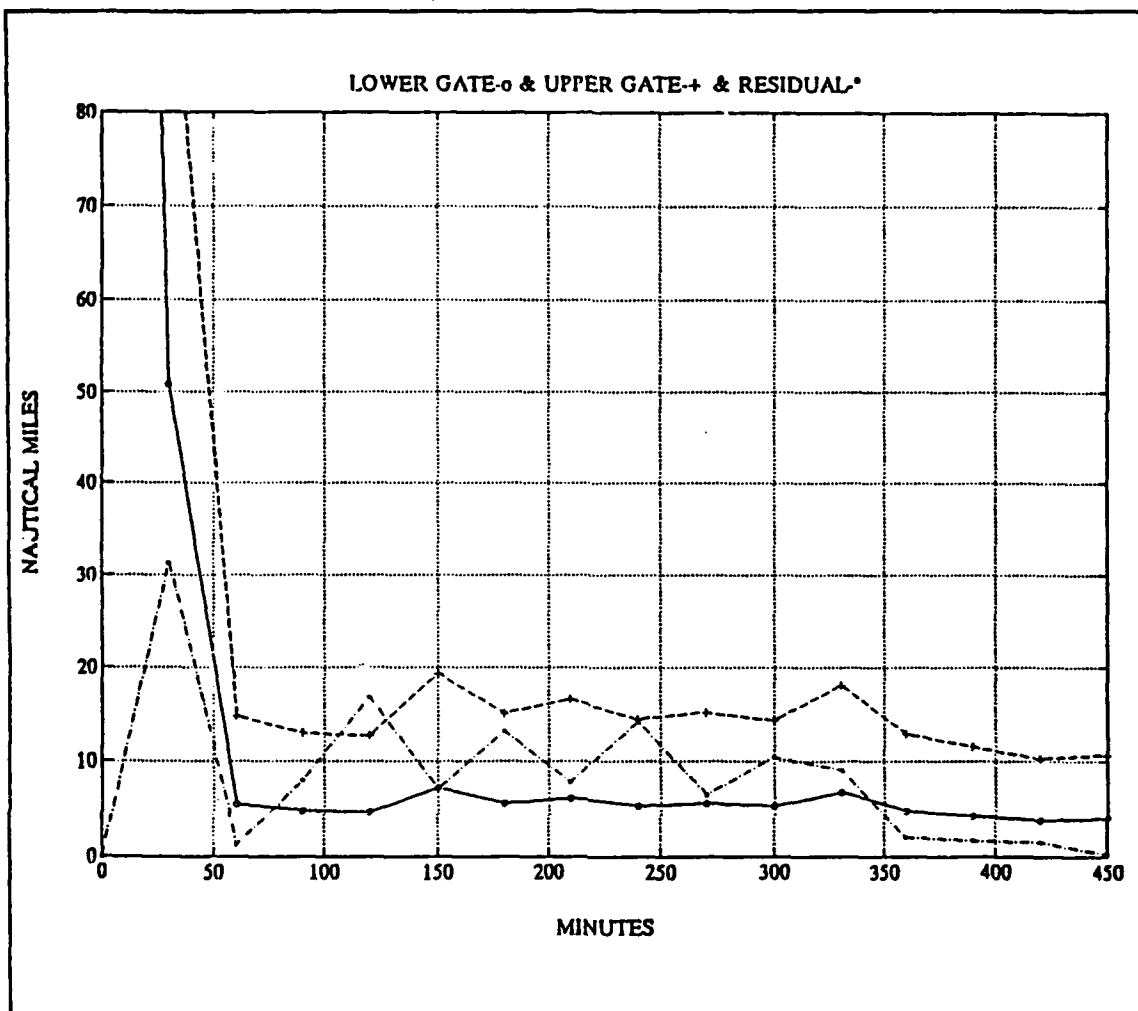
Figure 28. The Results of the Maneuver Detection Algorithm for Case #5
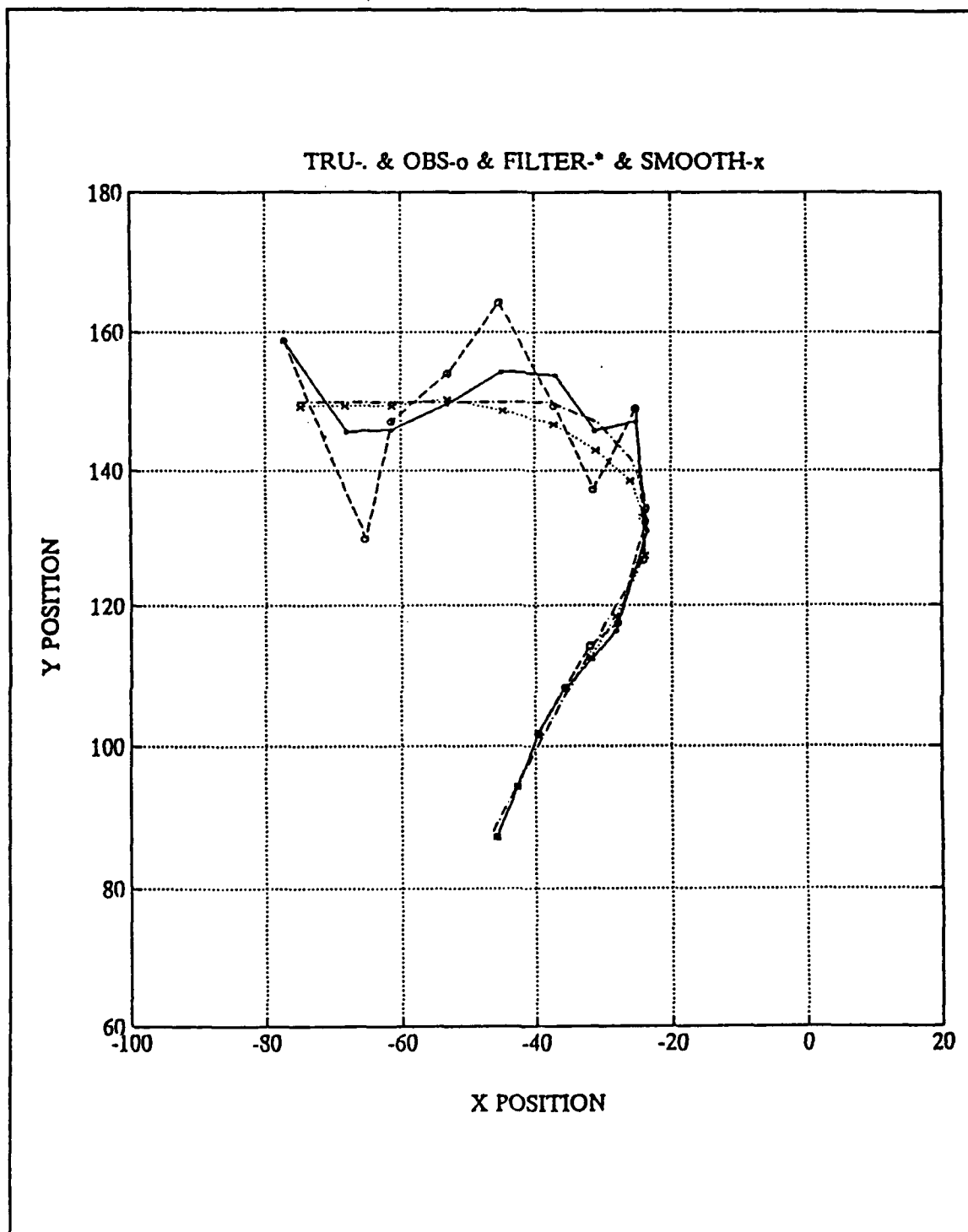
Figure 29.   The Overall Results for Case #5

G. CASE #6

This case is the same as Case #2, with the addition of noise to the measurements. The results of the extended Kalman filtering and smoothing are in Figure 30 and Figure 31.

From Figure 32, the maximum Kalman filter position error is 8 Nm at time equals 30 while the average filtered position is 3.3 Nm. The maximum errors are 3.5 Nm at time 420 and 6.5 Nm at time 450, which the filtered and smoothed errors had to be same, and the average error is 2.1 Nm for the smoothed errors. This case shows the general improvement in the filtered and smoothed estimates. The position accuracy increases by 47% with the extended Kalman filter and by 65% with the fixed-interval smoothing filter.

As seen in Figure 33, the residual values are in the maneuver zone at times 60, 180, 210, 240, 270 and 330 minutes. No maneuver detection occurs at times 60 and 330, since the residuals immediately following these times are out of the maneuver zone. The maneuver periods are detected from 210 to 270 minutes for the extended Kalman filter and from 180 to 270 minutes for the smoothing filter. The improvement in the accuracy of the position estimates is 49% due to the extended Kalman filter and 60% due to the smoothing filter.

Figure 33 also shows that the maneuver detection algorithm recognizes the observations at times 120, 300 and 420 as bad observations. As can be seen from Figure 30, the filtered positions are the projections of the previous estimates in time with no noise adaptation being made. For each of these times the filtered estimates are more accurate than the observed estimates and the smoothed estimates are the most accurate of all. The average improvement in the position estimate for these three observations is 56% for the extended Kalman filter and 83% for the smoothing filter. Figure 34 shows the overall tracking and smoothing results for this case.
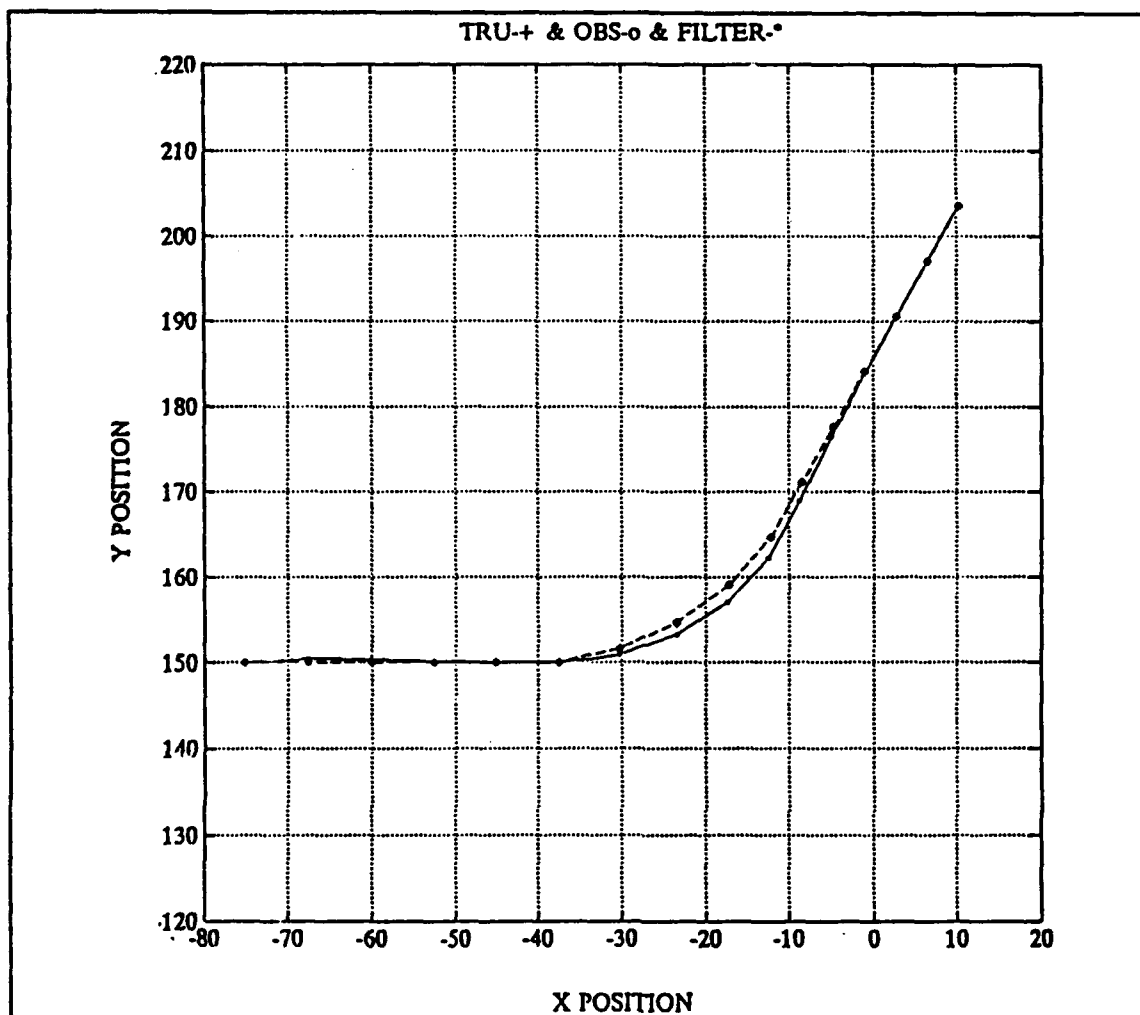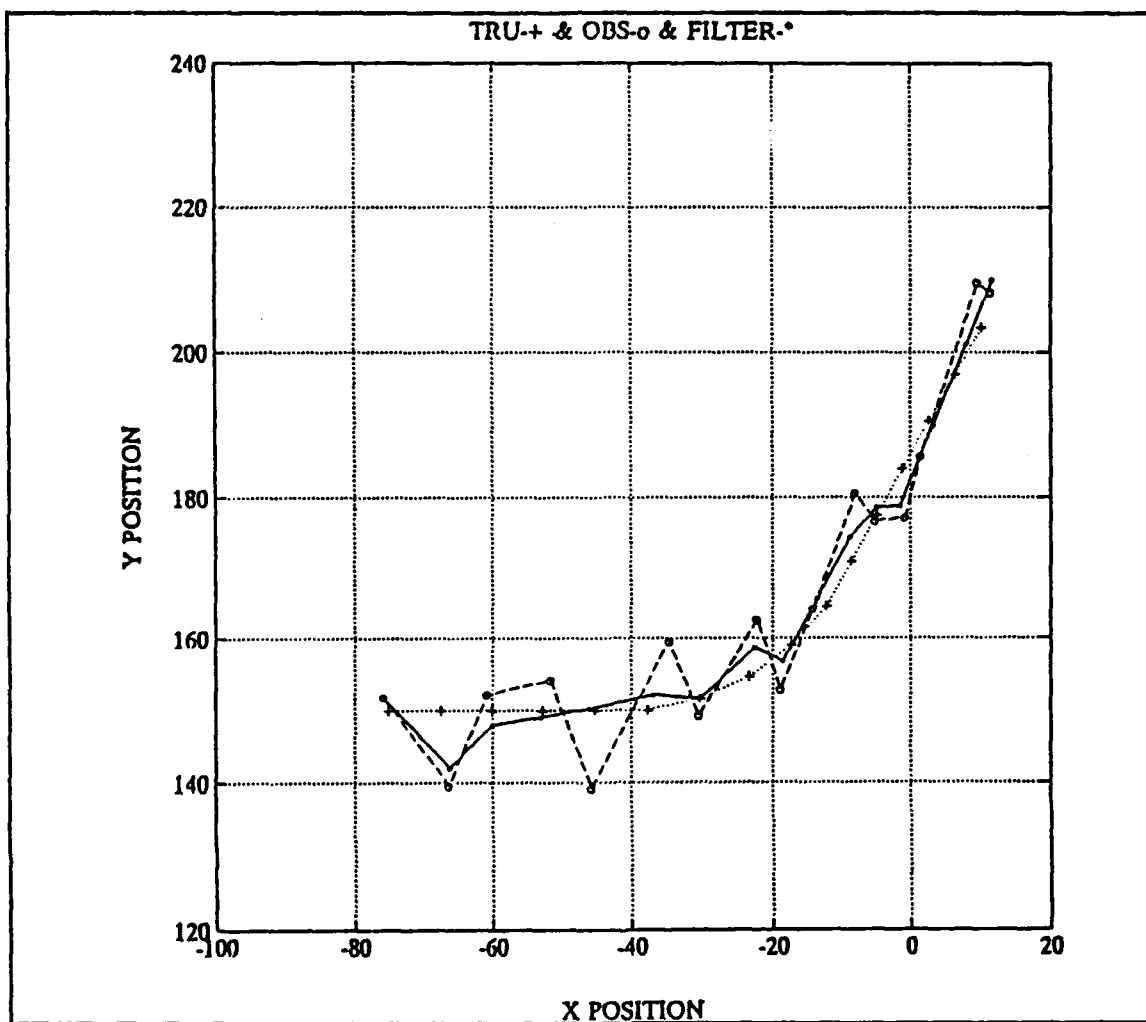
Figure 30. The Results of the Kalman Filter Tracking for Case #6
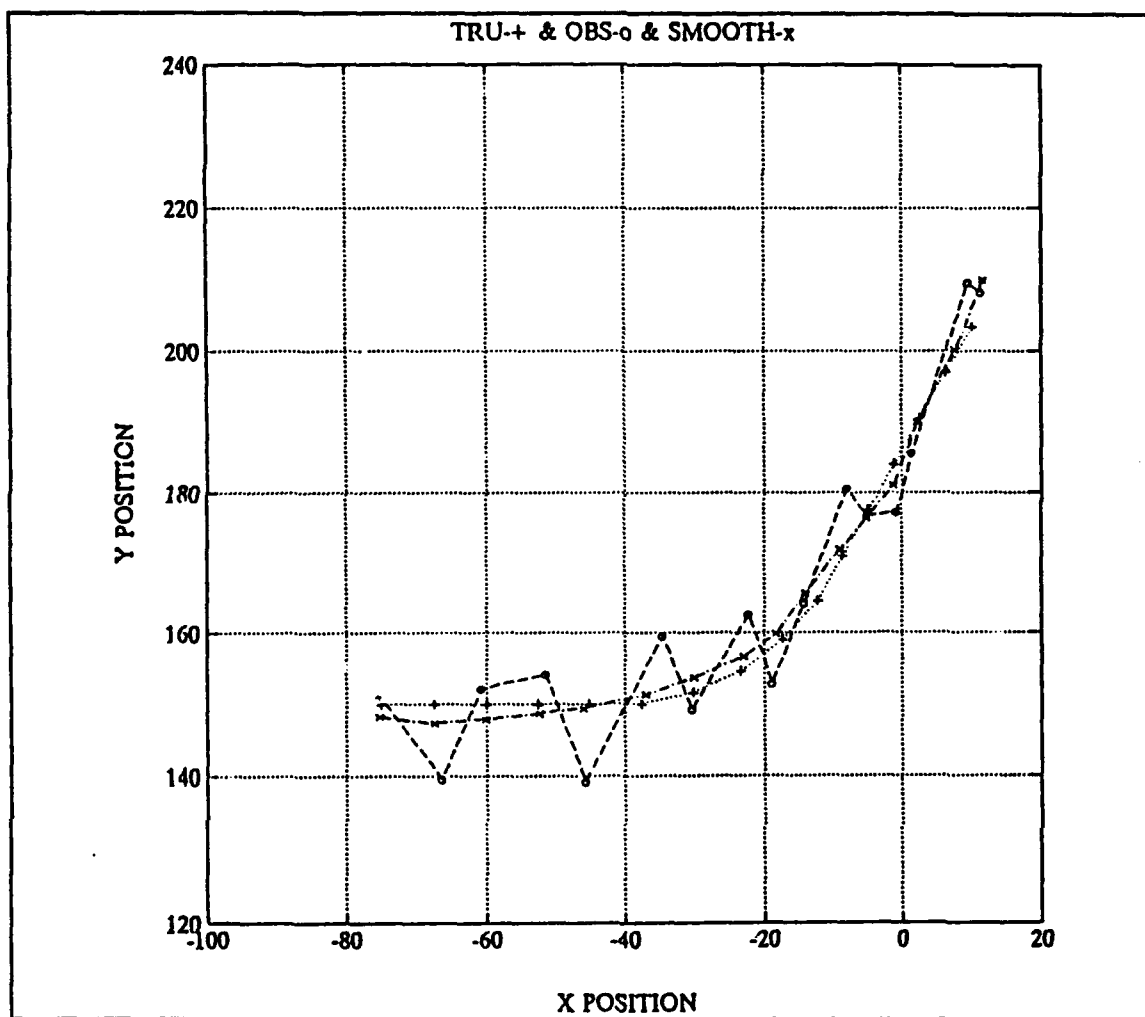
Figure 31.   The Results of the Fixed-Interval Smoothing for Case #6
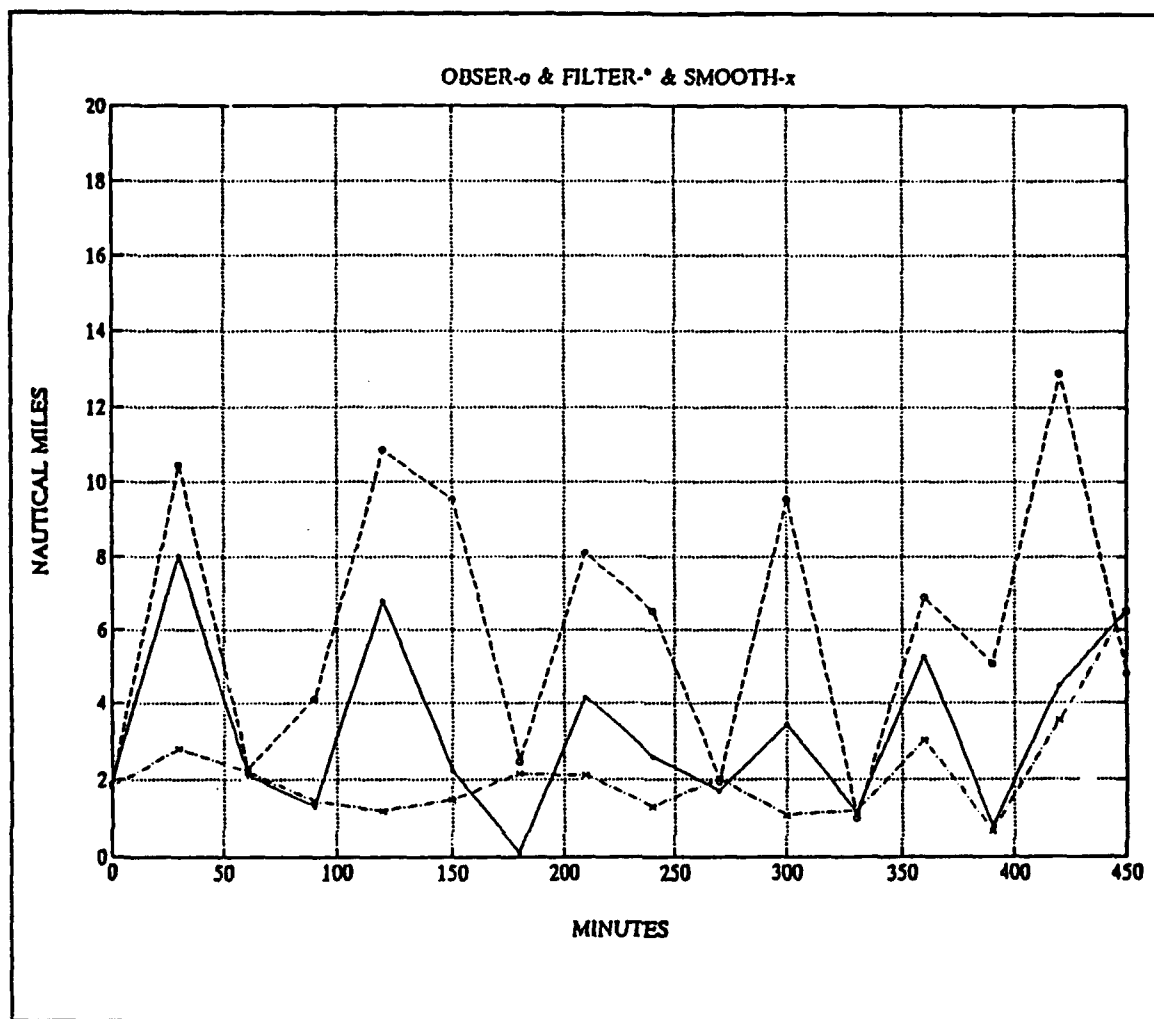
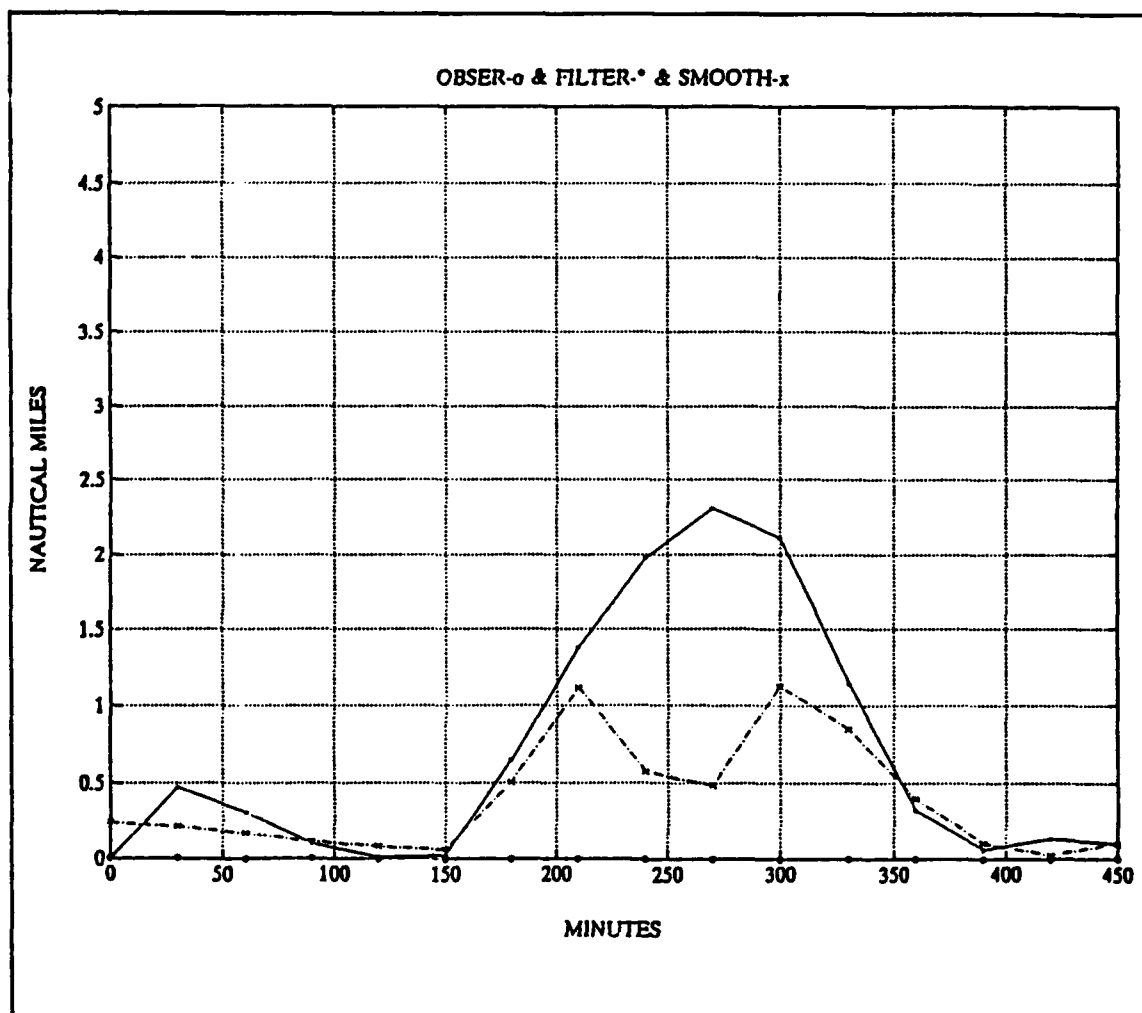Figure 32. The Position Errors for Case #6
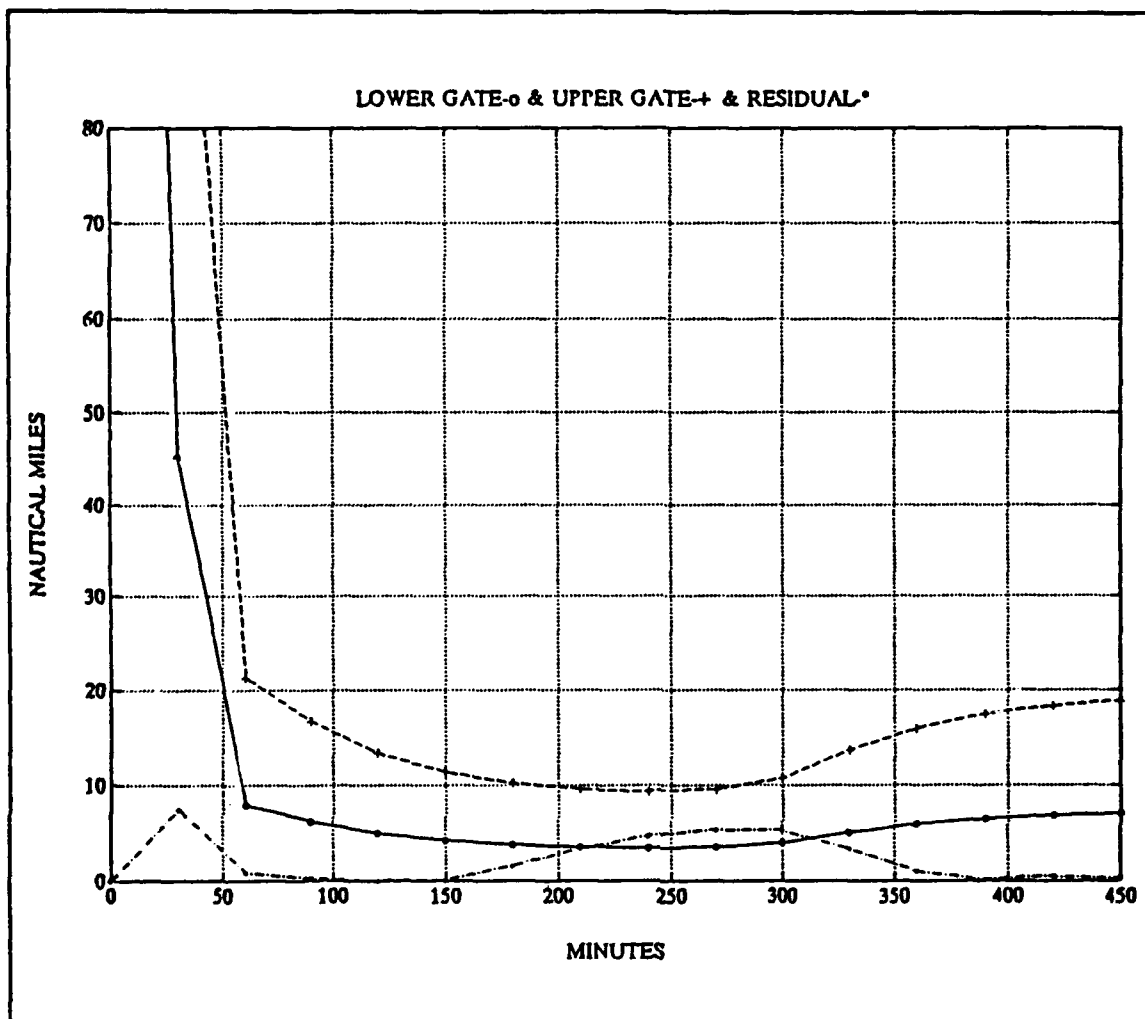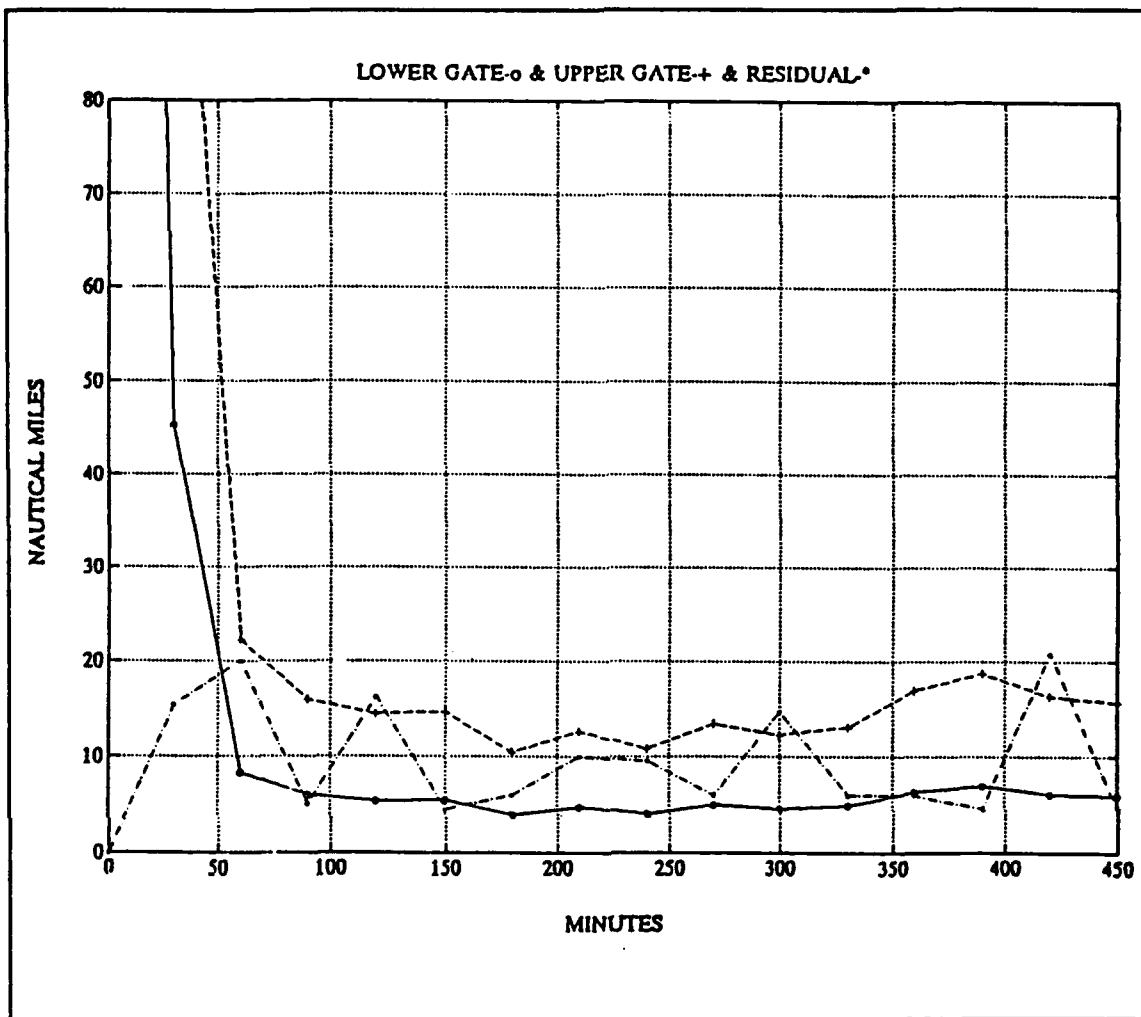
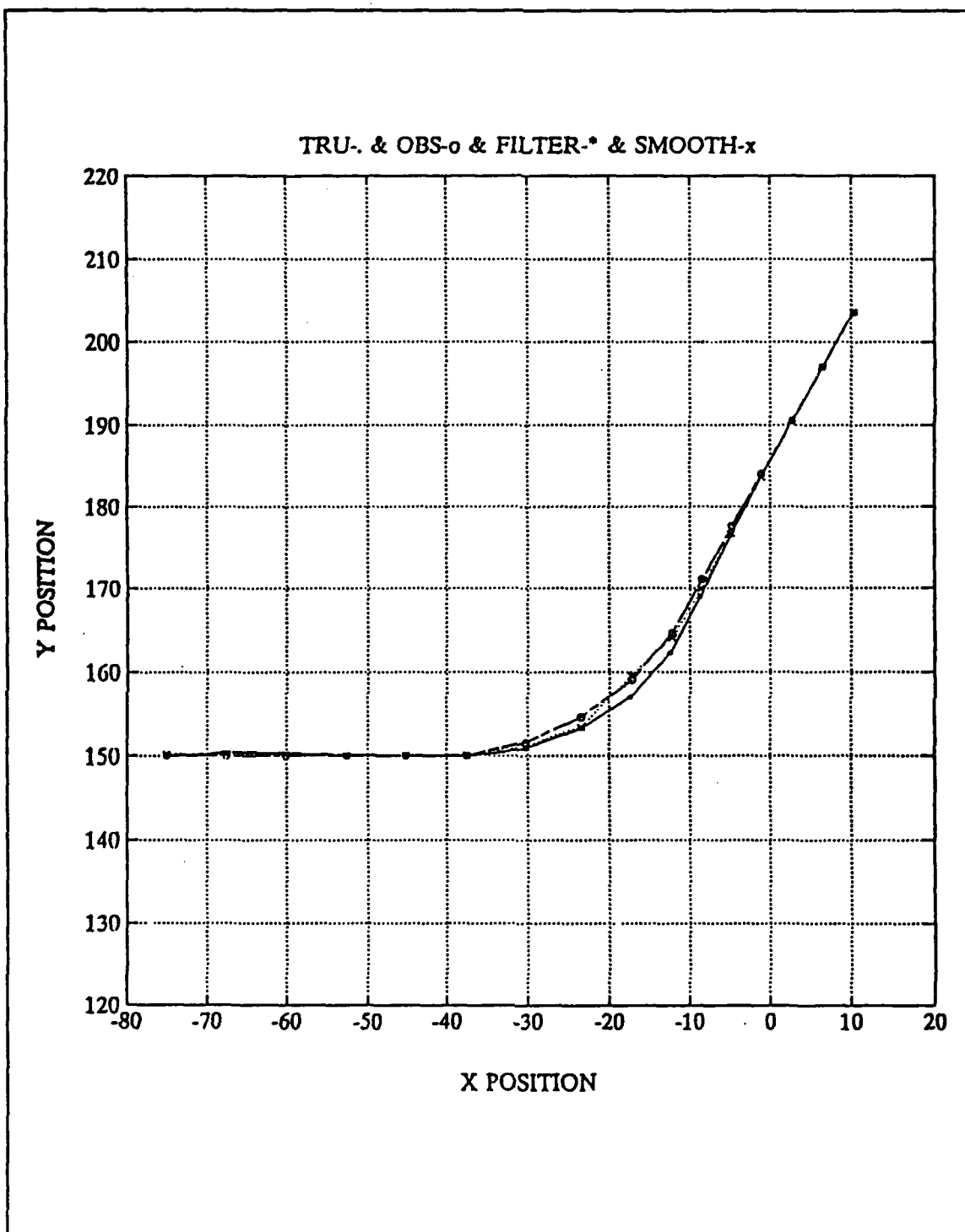**Figure 33.** The Results of the Maneuver Detection Algorithm for Case #6
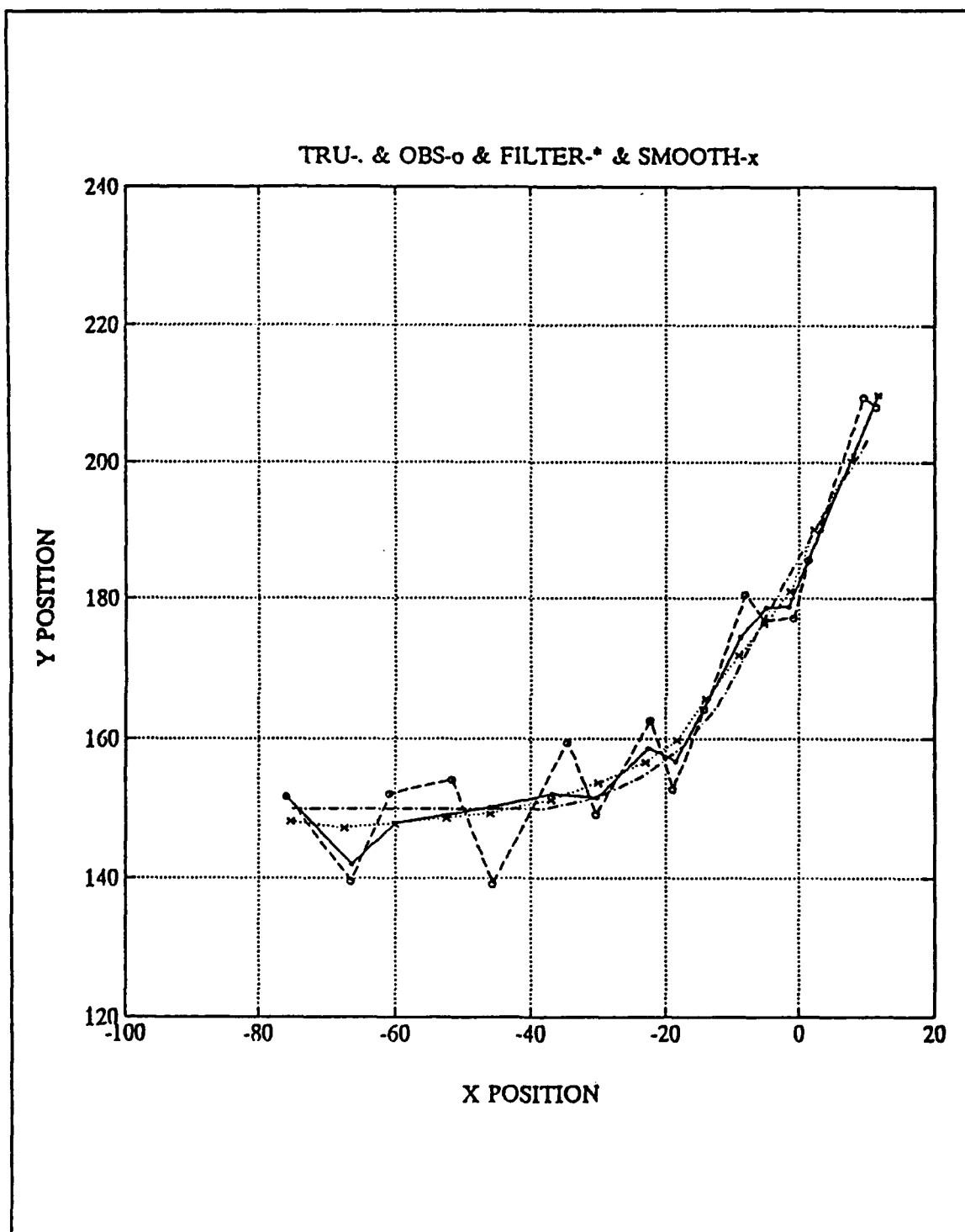
Figure 34.　The Overall Results for Case #6

H. CASE #7

This case depicts a 120° target maneuver away from the tracking ships. The filtered and smoothed tracks are in Figure 35 and Figure 36.

The observed, filtered and smoothed position errors are in Figure 37. The accuracy of the position estimates is increased by 47% with the extended Kalman filter and by 65% with the fixed-interval smoothing throughout the entire tracking period. The average position error due to the Kalman filter is 5.4 Nm while the average position error of the smoothed estimates is 2.3 Nm. The smoothed error is always less than 5 Nm with the exception of the last observation time (i.e. time 450) where the error is 5.4 Nm.

From Figure 38, the maneuver period is detected as times 180, 210, 240 and 270 for the extended Kalman filter and as times 150, 180, 210, 240 and 270 for the smoothing algorithm. During these periods, the accuracy in the position estimates is improved by 42% with the Kalman filter and by 72% with the smoothing. The observations of times 90 and 300 are recognized as bad observations. The Kalman filter improves the position accuracy by an average of 68% and the average improvement due to the smoothing routine is 90% for these two points. The overall results for this case are shown in Figure 39.

66
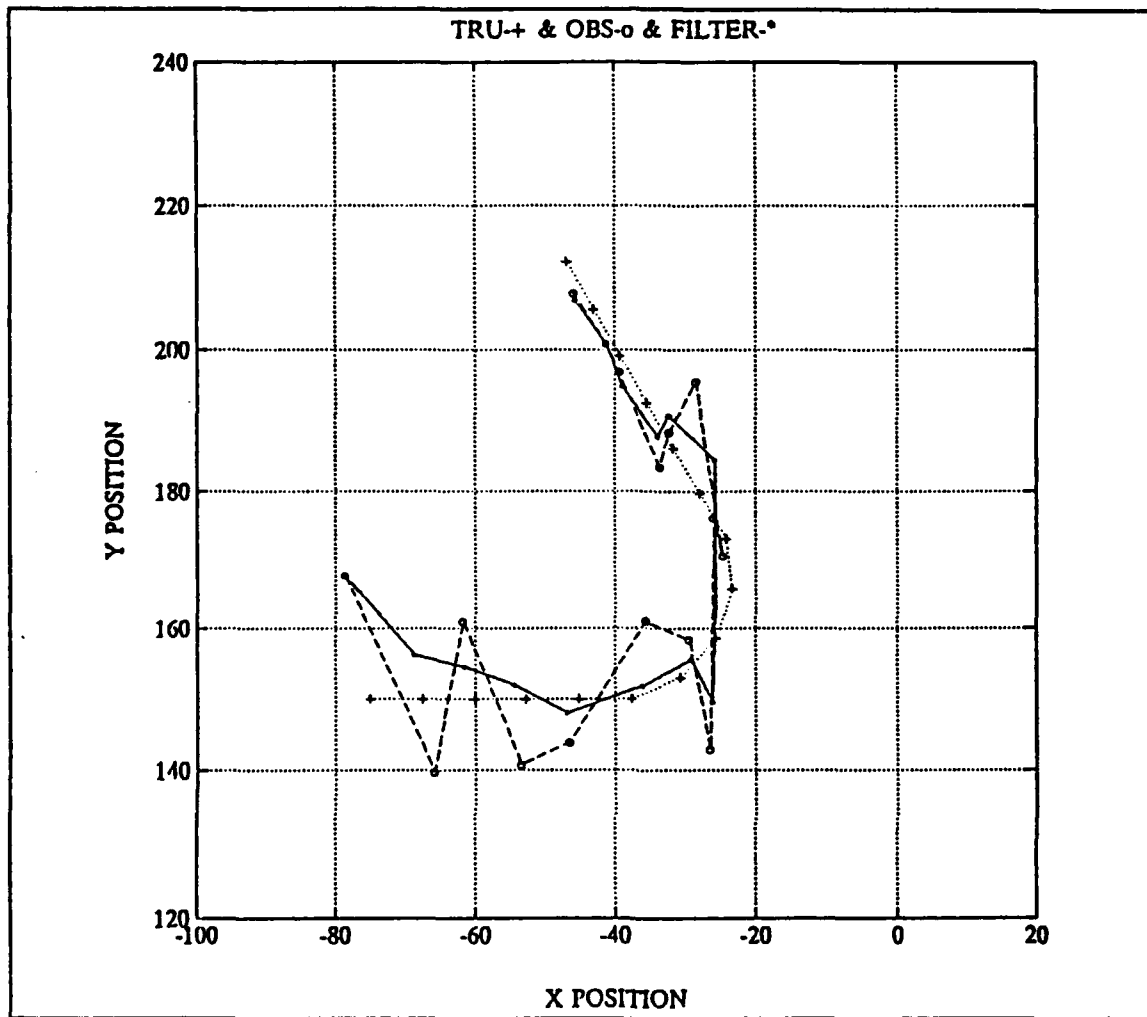
Figure 35. The Results of the Kalman Filter Tracking for Case #7
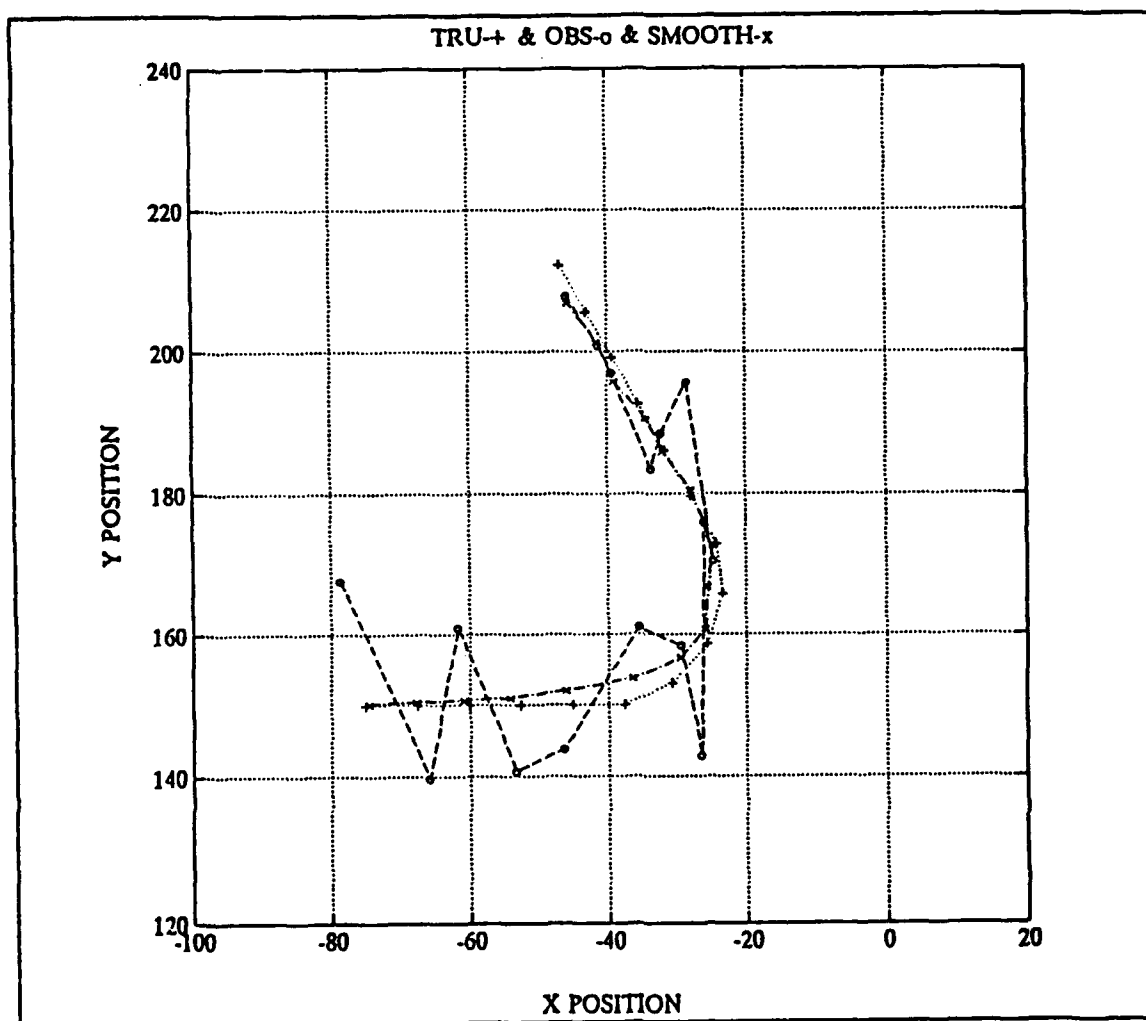
Figure 36.   The Results of the Fixed-Interval Smoothing for Case #7
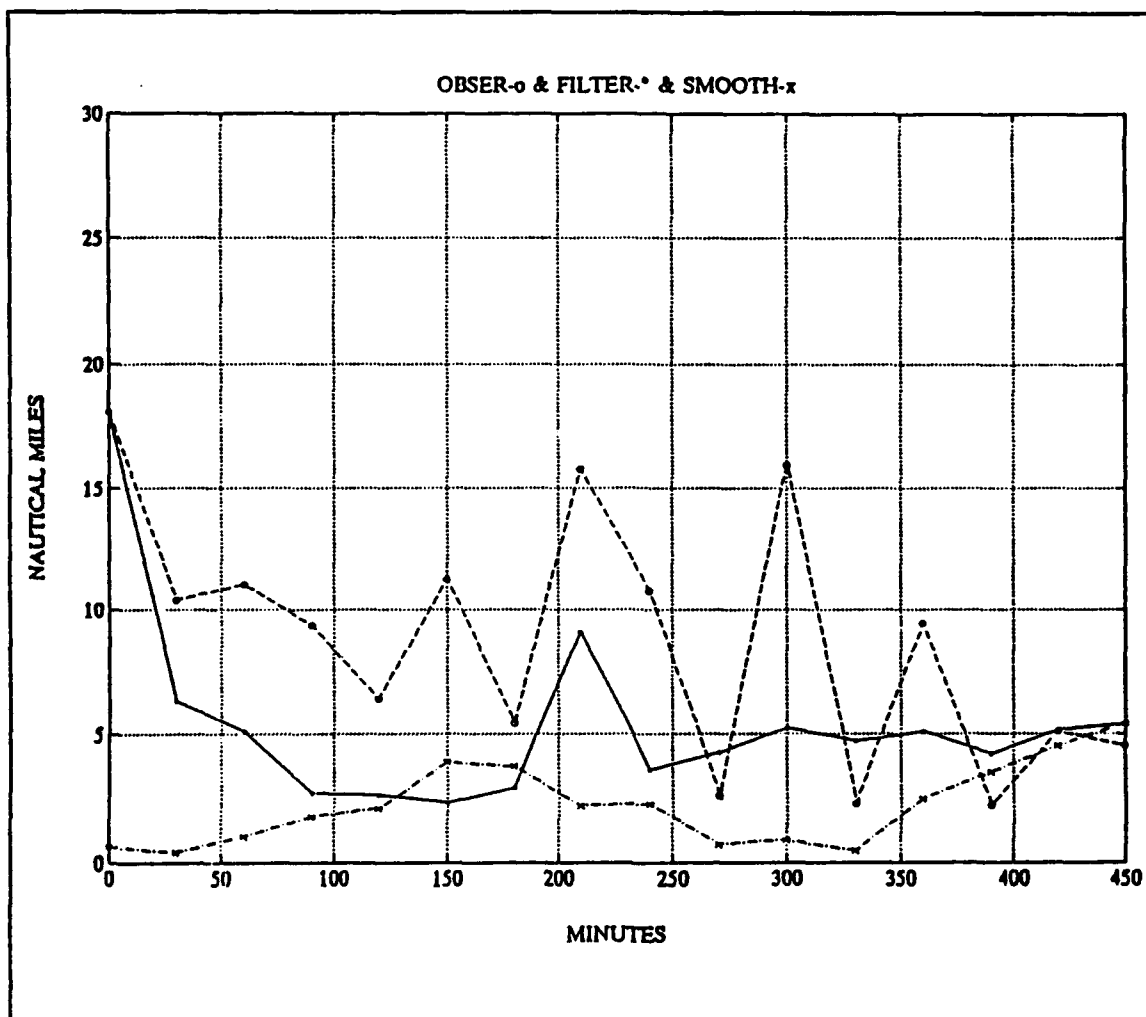
Figure 37. The Position Errors for Case #7

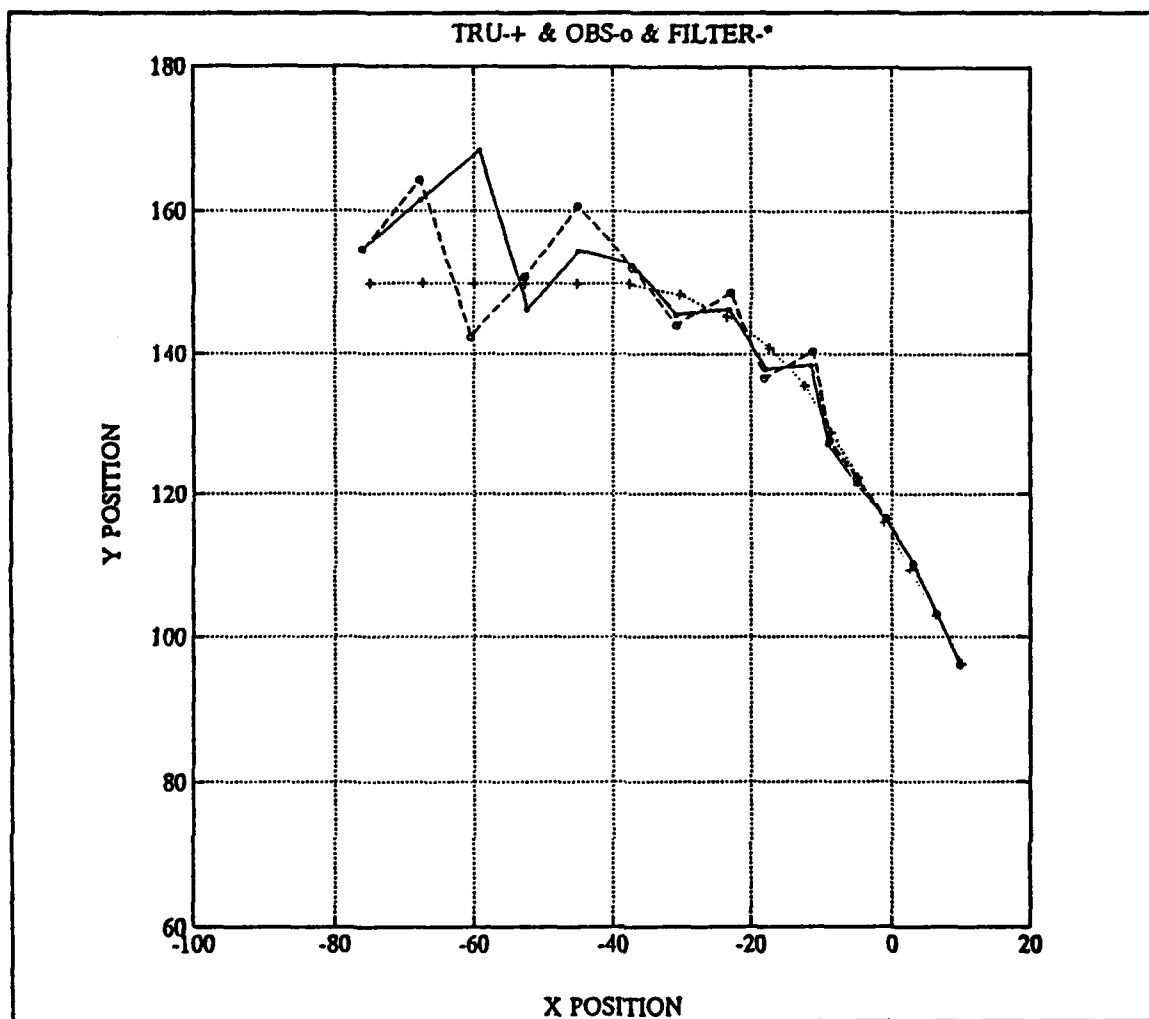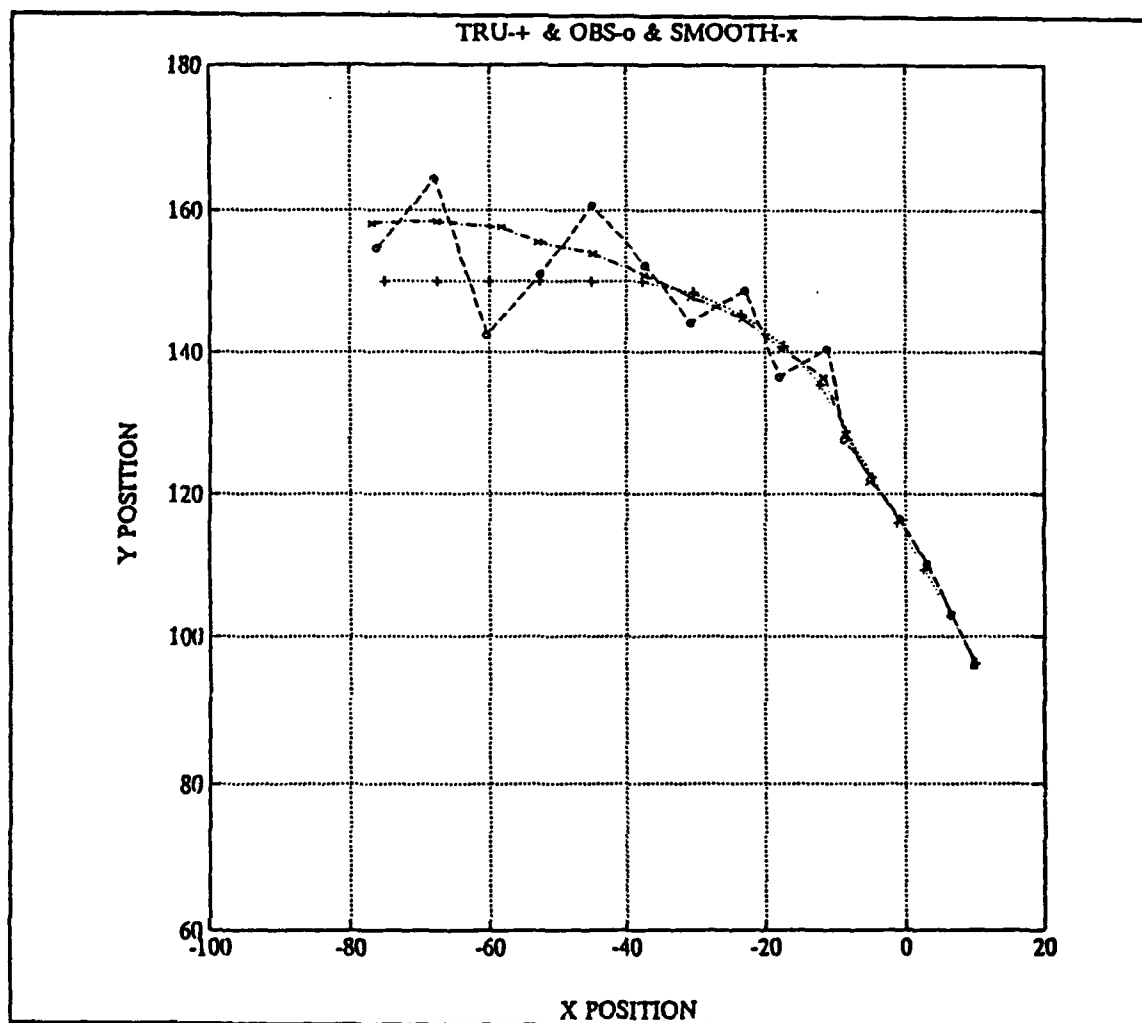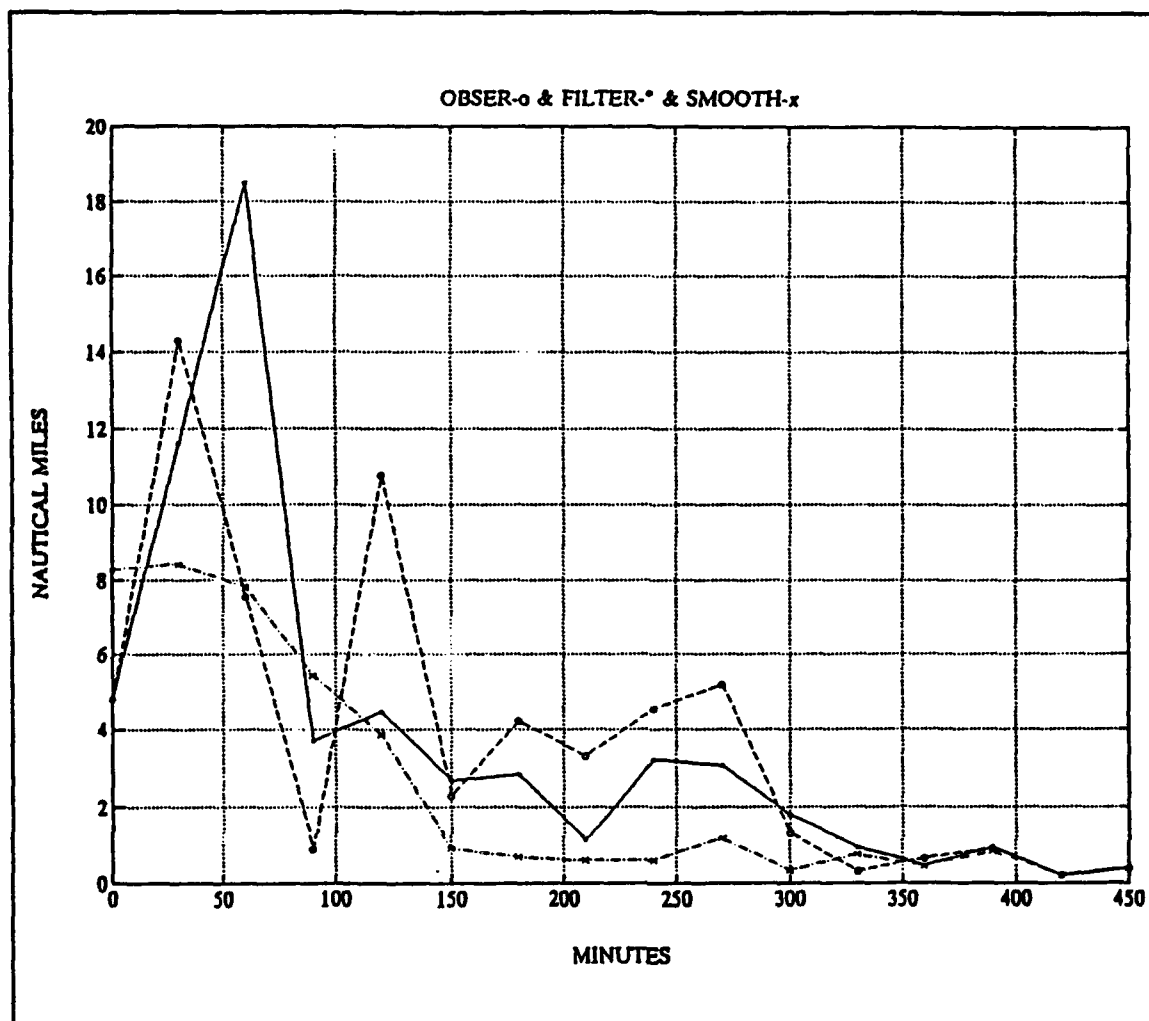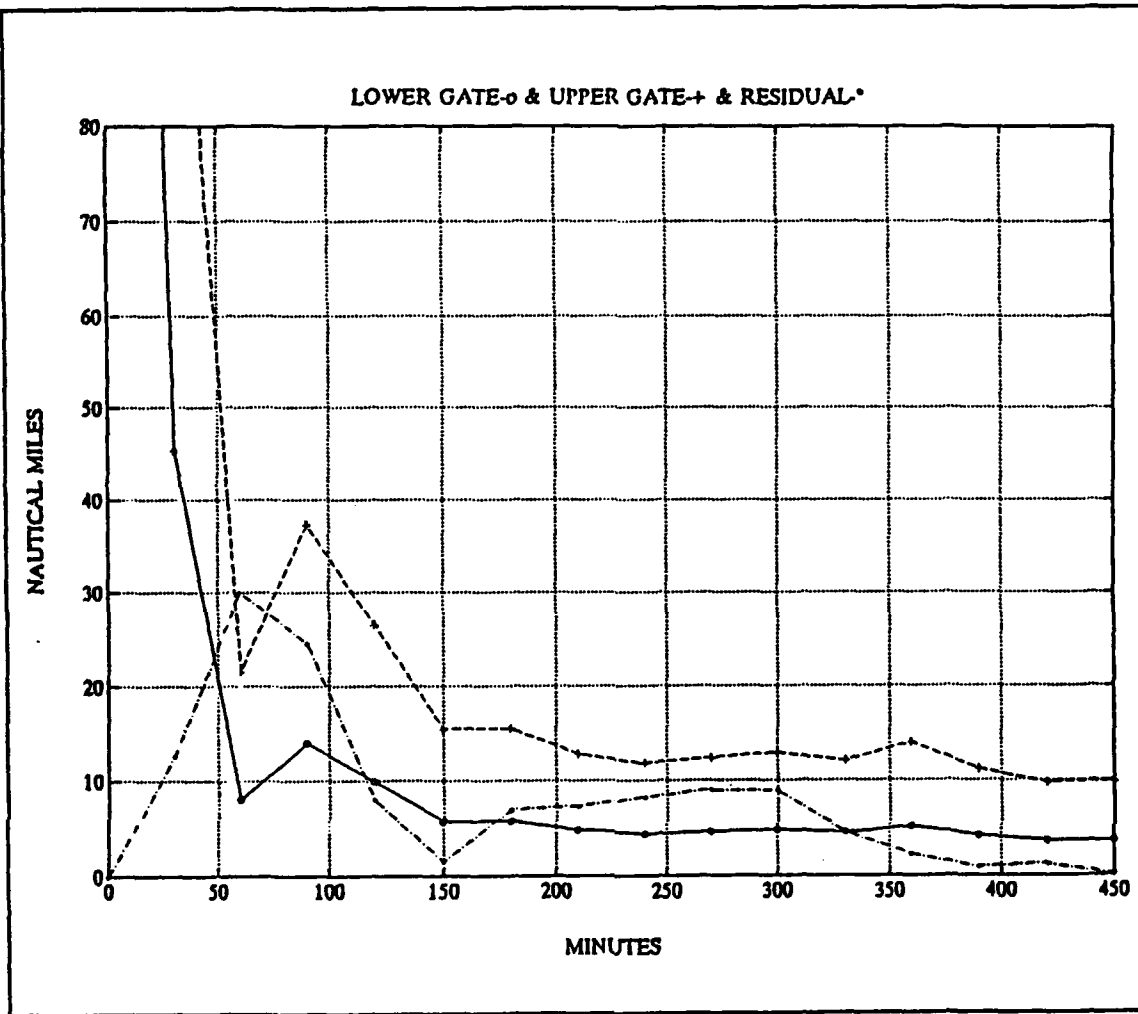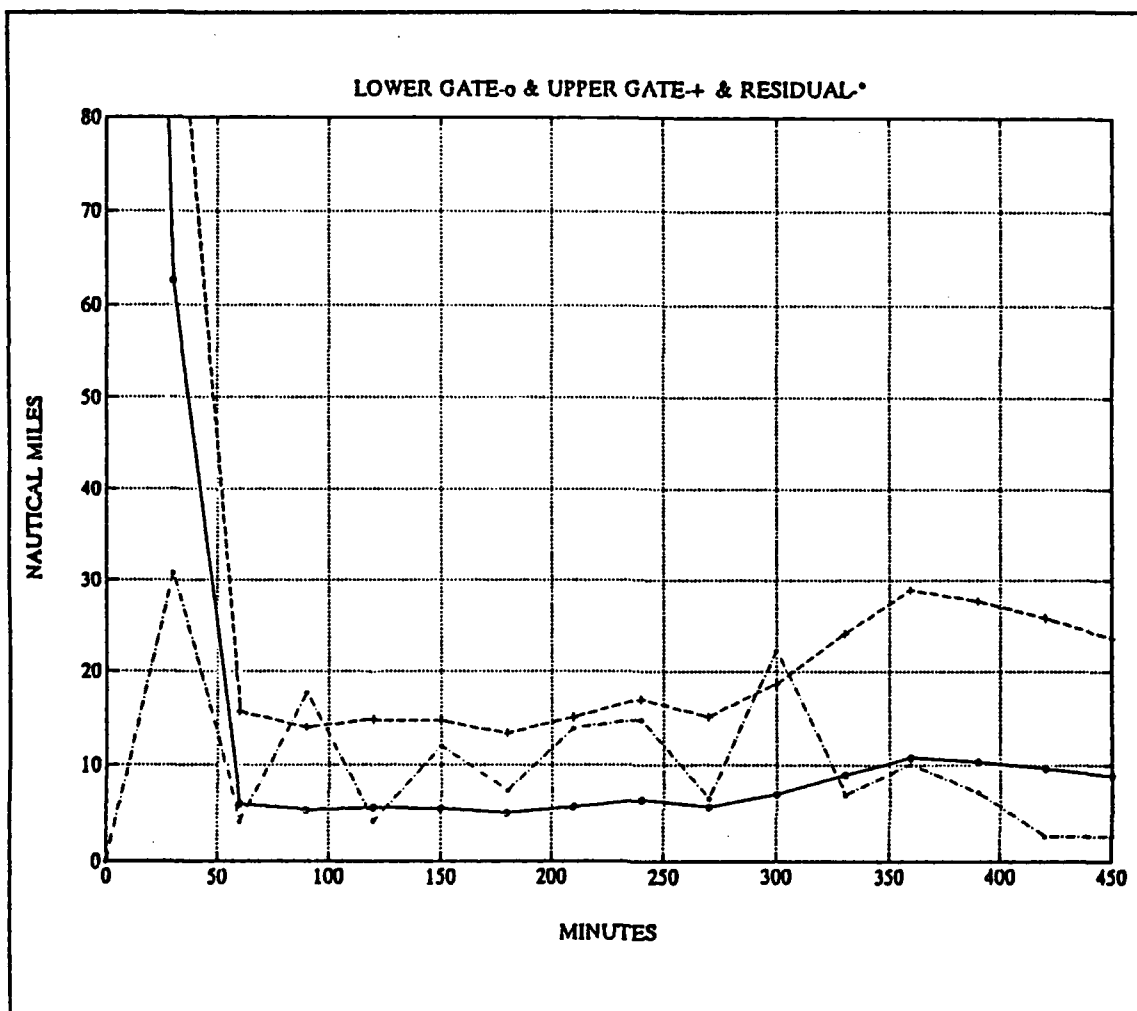Figure 38. The Results of the Maneuver Detection Algorithm for Case #7

70

TRU-. & OBS-o & FILTER-* & SMOOTH-x
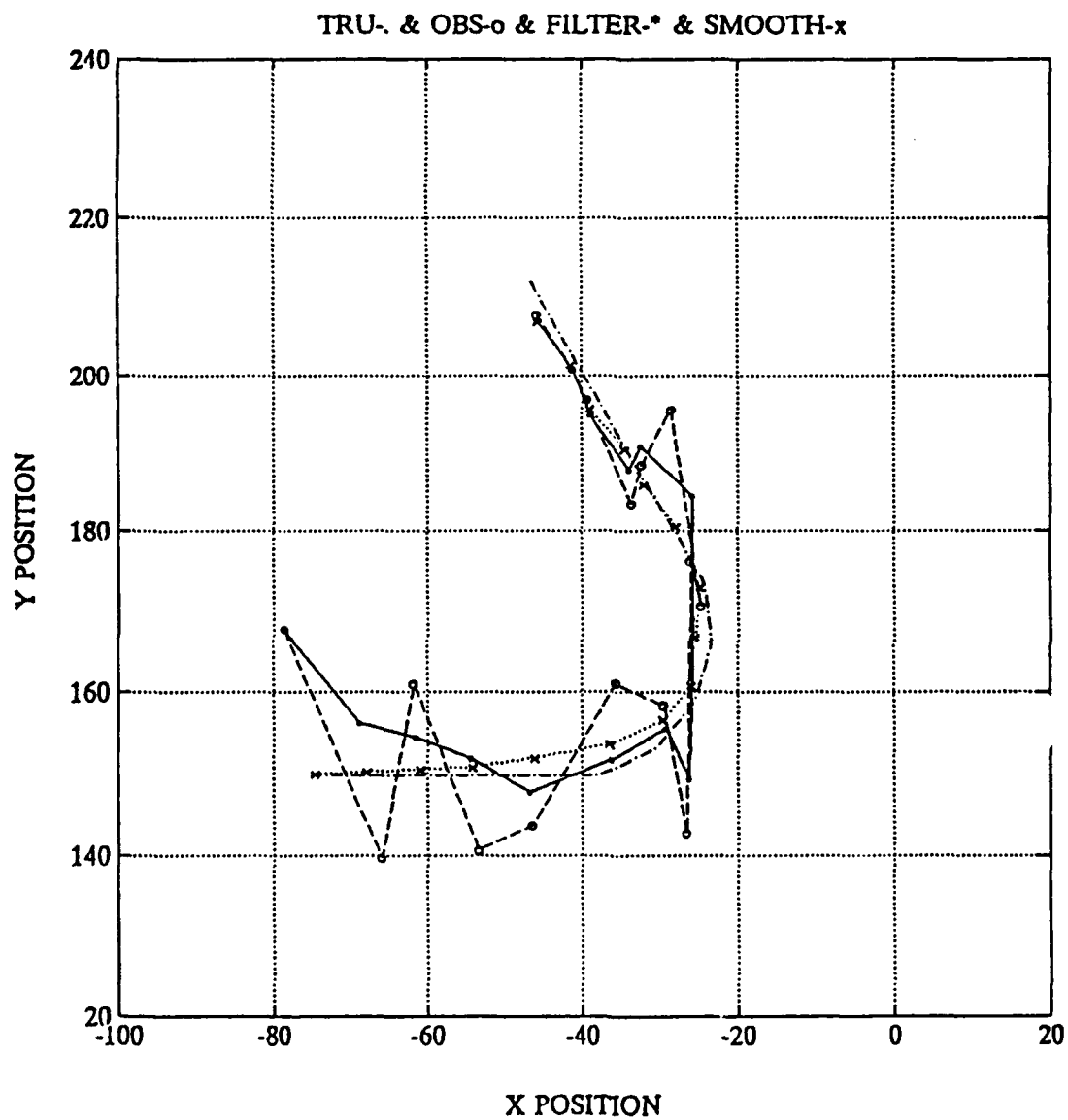
Figure 39.   The Overall Results for Case #7

71

VI. CONCLUSIONS

We have tried to improve the accuracy of the extended Kalman filter with a fixed-interval smoothing routine by implementing a new maneuver detection algorithm. Whereas maneuver detection algorithms are normally applied only to the extended Kalman filter, we apply this algorithm both to the extended Kalman filter and to the fixed-interval routine to adapt them both to unpredicted maneuvers of the target. We studied the effects of varying the state excitation matrix Q_K in the fixed-interval smoothing during the assumed maneuver periods. Several simulation cases were run and analyzed in order to test the performance of the algorithm.

Although some maneuver points were missed, the maneuver detection algorithm worked well during the simulations. The probabilities of a maneuver being detected for noise free and noisy cases are shown in Table 1 and Table 2. In order to obtain these probabilities a large number of simulations (i.e. 10) for both noise free and noisy cases were run on the IBM PC. Due to space constraints, just four representative runs each were presented in the previous chapter.. However, in Table 1 and Table 2 the results of all ten runs are shown for the fixed-interval smoothing routine only. To get the probabilities of a maneuver being detected in the extended Kalman filter, the reader must shift the numbers in both tables one cell right. In both Table 1 and Table 2, the maneuver was executed at N equal zero.

With a new maneuver detection technique, the fixed-interval smoothing routine improved the accuracy of the target's position estimates in all the simulation cases. This improvement was 35-/5% over the observed target positions and over 35-55% over the Kalman filter's estimates. Applying the new maneuver detection technique also improved the accuracy of the extended Kalman filter by 45-50% over the entire time in-

Table 1. Probabilities of a Maneuver Being Detected at Point N in a Noise Free Enviroinment for Fixed-Interval Smoothing Algorithm (Maneuver Executed at N = 0)

N = 0	N = 1	N = 2	N = 3	N = 4	N = 5
0%	0%	25%	75%	100%	100%

Table 2. Probabilities of a Maneuver Being Detected at Point N in a Noisy Enviroinment for a Fixed-Interval Smoothing Algorithm (Maneuver Executed at N = 0)

N = 0	N = 1	N = 2	N = 3
20%	80%	100%	100%

terval. Where the accuracy was most improved by this technique during the maneuver periods: here the accuracy increased by 30-60% using the extended Kalman filter and 60-80% using the the fixed-interval smoothing algorithm over the observed positions during the actual maneuver periods. And during the maneuver periods the smoothed estimates were 30-70% more accurate than the Kalman filter's estimates.

These significant improvements were obtained, in part, by using the time-varying values of the state excitation matrix Q_K. However, there is a disadvantage to this technique. Since this matrix is added to the predicted error covariance matrix $P_{K/K-1}$, high values of the matrix Q_K will cause the predicted error covariance matrix to grow boundlessly which will make the filter become unstable. Also, increasing the magnitude of the state excitation matrix in the fixed-interval smoothing algorithm makes the

73

smoothing filter estimates diverge to the extended Kalman filter estimates. This increases the need for a more accurate extended Kalman filter, since the accuracy of the smoothed estimates in this case depends to a large degree on the extended Kalman filter's estimates.

There are at least two areas which can be investigated to develop the tracking algorithm more fully. The first is research in new noise models. The model used was a white noise process. Although this model is relatively adequate for representing atmospheric noise over an extended time period, better models could be used which take into account random noise spikes, the lightning effects, of the atmospheric noise process. The second area is adapting the algorithm for multi-target tracking. Improving the ability of the algorithm to track and identify two or more targets would have great value in ship tracking and targeting problems.

APPENDIX A.   THE EXTENDED KALMAN FILTER WITH FIXED INTERVAL SMOOTHING ALGORITHM

```
C *** SHIPMANE.FOR ***
C**************************************************************
C*                                                          *
C*      THIS A EXTENDED KALMAN FILTER TRACKING ROUTINE WITH THE FIXED  *
C* INTERVAL SMOOTHING ALGORITHM.  THIS PROGRAM USES BEARINGS TAKEN  *
C* FROM TWO SENSOR SHIPS TO THE TARGET.  A NEW MANEUVER DETECTION  *
C* ROUTINE IS IMPLEMENTED.  THE NEW ALGORITHM TO DERIVE THE STATE  *
C* EXCITATION MATRIX Q IS ALSO DEVELOPED.  TO RUN THE PROGRAM:  *
C*                                                          *
C*      1) RUN THE PROGRAM <RAWDATA.FOR> LOCATED IN APPENDIX B TO  *
C*         PRODUCE THE RAW DATA.                            *
C*                                                          *
C*      2) RUN THE <SHIPMANE.FOR>                           *
C*                                                          *
C*   THE OUTPUTS OF THE PROGRAM STORED IN THE FOLLOWING FILES:  *
C*                                                          *
C*      1) THRDATA   = INCLUDES THE THERESHOLD VALUES OF THE  *
C*                     MANEUVER GATES AND RESIDUAL CALCULATED  *
C*                     FOR EACH OBSERVATION.                *
C*                                                          *
C*      2) CIRCDATA  = INCLUDES THE REQUIRED DATA TO DRAW THE  *
C*                     MANEUVER GATES AS A CONCENTRIC CIRCLES  *
C*                     AROUND THE PREDICTED FILTER ESTIMATES.  *
C*                                                          *
C*      3) BEGINDATA = INCLUDES THE FIRST POINTS, IGNORED BY  *
C*                     THE EXTENDED KALMAN FILTER, OF THE TARGET  *
C*                     MANEUVERS TO BE USED BY THE FIXED-INTERVAL  *
C*                     SMOOTHING ALGORITHM.                 *
C*                                                          *
C*      4) MANEUDATA = INCLUDES THE DETECTED MANEUVER POINTS.  *
C*                                                          *
C*      5) TRUDATA   = INCLUDES THE ACTUAL POSITION OF THE TARGET  *
C*                     FOR EACH OBSERVATION TIME.           *
C*                                                          *
C*      6) FILDATA   = INCLUDES THE EXTENDED KALMAN FILTER'S POSITION  *
C*                     ESTIMATES ALONG WITH THE OBSERVED POSITIONS,  *
C*                     KALMAN FILTER ERROR AND OBSERVATION ERROR.  *
C*                                                          *
C*      7) SMDATA    = INCLUDES THE SMOOTHED POSITION ESTIMATES AND  *
C*                     SMOOTHING ERROR.                     *
C*                                                          *
C*   TO GET THE GRAPHIC RESULTS:                            *
C*                                                          *
C*      1) COPY THE FILES TRUDATA, FILDATA, SMDATA AND MANEUDATA INTO  *
C*         THE MATLAB SUB-DIR.                              *
C*                                                          *
C*      2) RUN THE PROGRAM <SHIPTR.M> IN THE MATLAB SUB-DIR.  THE  *
C*         GRAPHIC RESULTS WILL BE STORED IN THE META FILE SHIPTR.MET.  *
C**************************************************************
```

```
C  ***VARIABLE DEFINITIONS***

C       AK          =       SMOOTHING FILTER GAIN MATRIX
C       AKT         =       TRANSPOSE OF AK
C       BD          =       BAD OBSERVATION INDICATOR
C       BOC         =       BAD OBSERVATION COUNTER WHICH PROVIDES
C                           THAT ONLY THE FIRST OF TWO CONCECUTIVE
C                           BAD OBSERVATIONS WILL BE RECOGNIZED
C       BRG         =       MEASURED TARGET BEARING IN RADIANS
C       BRKKM1      =       PREDICTED TARGET BEARING MEASUREMENT
C                           IN RADIANS, BRG(K/K-1)
C       DBRG        =       MEASURED TARGET BEARING IN DEGREES
C       DT          =       TIME DELAY BETWEEN OBSERVATIONS,
C                                   T(K) - T(K1)
C       DTOR        =       DEGREE TO RADIAN CONVERSION FACTOR
C       FAC1        =       RECIPROCAL OF VARE
C       G           =       KALMAN GAIN VECTOR
C       H           =       MEASUREMENT MATRIX
C       HDG         =       TARGET HEADING IN DEGREES BY KALMAN FILTER
C       HT          =       TRANSPOSE OF H
C       I           =       COUNTER
C       IMAT        =       4 X 4 IDENTITY MATRIX
C       J           =       COUNTER
C       K           =       ITERATION INTERVAL
C       PDIFF       =       POSITION DIFFERENCE BETWEEN OBSERVED AND
C                           PREDICTED STATE ESTIMATES
C                                   | Z(K) - X(K/K-1) |
C       PHI         =       DISCRETE-TIME STATE TRANSITION MATRIX
C       PHIT        =       TRANSPOSE OF PHI
C       PKK         =       ESTIMATION ERROR COVARIANCE MATRIX, P(K/K)
C       PKKS        =       SMOOTHED ERROR COVARIANCE MATRIX
C       PKKM1       =       PREDICTED ESTIMATION ERROR COVARIANCE
C                           MATRIX, P(K+1/K)
C       PKKM1S      =       PREDICTED ERROR COVARIANCE MATRIX FOR
C                           SMOOTHING, P(K+1/K)
C       IPKKM1S     =       INVERSE OF PKKM1S
C       PSS         =       ERROR COVARIANCE MATRIX FOR SMOOTHING, P(K/K)
C       PX          =       POSITION DIFFERENCE IN X DIRECTION BETWEEN
C                           OBSERVED AND PREDICTED STATE ESTIMATES
C                                   | ZX - X(K/K-1)(1,1) |
C       PY          =       POSITION DIFFERENCE IN Y DIRECTION BETWEEN
C                           OBSERVED AND PREDICTED STATE ESTIMATES
C                                   | ZY - X(K/K-1)(3,1) |
C       Q           =       STATE EXCITATION MATRIX
C       R           =       MEASUREMENT NOISE COVARIANCE
C       RANGE       =       DISTANCE FROM SENSOR TO A PRIORI TARGET
C                           POSITION
C       RTOD        =       RADIAN TO DEGREE CONVERSION FACTOR
C       SHDG        =       TARGET HEADING IN DEGREES BY SMOOTHING
C       SPD         =       TARGET SPEED IN KNOTS BY KALMAN FILTER
C       SPKKM1      =       STORE THE INITIAL ERROR COVARIANCE, P(0/-1)
C       SSPD        =       TARGET SPEED IN KNOTS BY SMOOTHING
C       SXPOS       =       SMOOTHED TARGET POSITION IN X DIRECTION
C       SYPOS       =       SMOOTHED TARGET POSITION IN Y DIRECTION
C       TEMP        =       TEMPORARY STORAGE MATRICES USED IN MATRIX
```

```
C                               OPERATIONS
C       TIMEX         =         IMPOSSIBLY HIGH CONSTANT FOR DETERMINING THE
C                               FIRST POINT OF THE MANEUVER WHICH IS IGNORED
C                               BY THE KALMAN FILTER AND ACCOUNT BY THE SMOOTHING
C                               FILTER
C       TIMEXLO       =         VARIABLE FOR STORING THE MANEUVER POINTS
C       TL            =         THRESHOLD VALUE FOR THE LOWER MANEUVER GATE
C       TLS           =         VECTOR VARIABLE FOR STORING THE THRESHOLD
C                               VALUES OF THE LOWER MANEUVER GATE
C       TU            =         THRESHOLD VALUE FOR THE UPPER MANEUVER GATE
C       TUS           =         VECTOR VARIABLE FOR STORING THE THRESHOLD
C                               VALUES OF THE UPPER MANEUVER GATE
C       VARE          =         VARIANCE OF RESIDUALS PROCESS
C       XDIFF         =         DISTANCE IN X DIRECTION FROM SENSOR TO A
C                               PRIORI TARGET POSITION
C       XKK           =         ESTIMATED TARGET STATE VECTOR, X(K/K)
C       XKKS          =         SMOOTHED TARGET STATE VECTOR
C       XKKM1         =         PREDICTED TARGET STATE VECTOR, X(K/K-1)
C       XKKM1S        =         PREDICTED TARGET STATE VECTOR FOR SMOOTHING, X(K+1/K
C       XPOS          =         KALMAN FILTERED TARGET POSITION IN X DIRECTION
C       XS            =         SENSOR POSITION IN X DIRECTION
C       XSS           =         TARGET STATE VECTOR FOR SMOOTHING, X(K/K)
C       XT            =         TRUE TARGET POSITION IN X DIRECTION
C       YDIFF         =         DISTANCE IN Y DIRECTION FROM SENSOR TO A
C                               PTIORI POSITION
C       YPOS          =         KALMAN FILTERED TARGET POSITION IN    DIRECTION
C       YS            =         SENSOR POSITION IN Y DIRECTION
C       YT            =         TRUE TARGET POSITION IN Y DIRECTION
C       ZX            =         OBSERVED POSITION IN X DIRECTION
C       ZY            =         OBSERVED POSITION IN Y DIRECTICN

C *** VARIABLE DECLARATIONS ***

        REAL*4 XKK(4,1),XKKM1(4,1),PHI(4,4),SXPOS,SYPOS,HDG,PDIFF
        REAL*4 H(1,4),G(4,1),TEMP1(1,4),TEMP2(1,1),TEMP3(4,1),ZT
        REAL*4 TEMP4(4,4),TEMP5(4,4),PKK(4,4),PKKM1(4,4),HT(4,1)
        REAL*4 LXKK(4,1),LPKK(4,4),XS(10),YS(10),DBRG(10),BRG(10)
        REAL*4 TEMP6(4,4),PHIT(4,4),IMAT(4,4),XT,YT,SHDG,XPL(100)
        REAL*4 VARE(2),E(2),G11,G13,G21,G23,Q(4,4),SSPD(100),MC
        REAL*4 DT,XDIFF,YDIFF,RANGE,XS1,YS1,BRG1,BRKKM1,YPL(100)
        REAL*4 OBSERR(200),FAC1,SIGTHT,SIGVT,R,RTOD,SPD(100),BD
        REAL*4 XS2,YS2,BRG2,ZX,ZY,DTOR,TRKERR(100),TL,TU,SX,SY
        REAL*4 XNNM1(4,1),XSS(4,1),XKKM1S(4,1),THETA,PY,PX,PD
        REAL*4 PNNM1(4,4),PSS(4,4),PKKM1S(4,4,100),IPKKM1S(4,4)
        REAL*4 AK(4,4),AKT(4,4),STRKERR(100),DTS(100),SPKKM1(4,4)
        REAL*4 TEMP1S(4,4),TEMP2S(4,1),TEMP3S(4,1),TH1(4,1),YPOS
        REAL*4 TEMP4S(4,4),TEMP5S(4,4),TEMP6S(4,4),TH2(4,4),XPOS
        REAL*4 XKKS(4,1,100),PKKS(4,4,100),TLS(100),TUS(100),DR(100)
        REAL*4 XPU(100),YPU(100),BOC

        INTEGER*4 TIME,TIMEP(100),NP,TIMEX,TIMEXB(100),TIMEXLO(100)

C *** OPEN OUTPUT DATA FILES ***
        OPEN(UNIT=2,FILE='TRKDATA.DAT',STATUS='OLD')
        OPEN(UNIT=3,FILE='THRDATA.DAT',STATUS='NEW')
        OPEN(UNIT=4,FILE='CIRCDATA.DAT',STATUS='NEW')
```

77

```
          OPEN(UNIT=6,FILE='MANEUDATA.DAT',STATUS='NEW')
          OPEN(UNIT=7,FILE='TRUDATA.DAT',STATUS='NEW')
          OPEN(UNIT=8,FILE='FILDATA.DAT',STATUS='NEW')
          OPEN(UNIT=9,FILE='SMDATA.DAT',STATUS='NEW')

C *** RADIAN/DEGREE CONVERSION FACTORS ***

          RTOD=57.29577951
          DTOR=0.01745293

C *** COMPUTE 4X4 IDENTITY MATRIX ***

          DO 5 I=1,4
          DO 5 J=1,4
          IF (I.EQ.J) THEN
                  IMAT(I,J)=1.0
          ELSE
                  IMAT(I,J)=0.0
          ENDIF
5         CONTINUE

C *** INITIALIZE TIME AND MANEUVER DETECTION ALGORITHM COUNTERS ***

          TIMEM1=0
          NP=1
          TIMEX=5000
BOC=0.0

C *** COMPUTE BEARING MEASUREMENT COVARIANCE ***
C      BEARING ERROR STANDARD DEVIATION = 3 DEGREES

          R=(3*DTOR)**2


C ***********************************************************************
C *            THIS WHERE THE EXTENDED KALMAN FILTERING STARTS          *
C ***********************************************************************

C *** READ IN OBSERVATION PACKET (TIME, # OF SENSORS) ***
C      DT=TIME(K)-TIME(K-1)

          WRITE(*,*)'EXTENDED KALMAN FILTERING NOW STARTS'
          WRITE(*,*)'***===========================***'

810       READ(2,1001,END=800)TIME,XT,YT,XS(1),YS(1),DBRG(1),
     *            XS(2),YS(2),DBRG(2)
1001          FORMAT(I4,8F9.4)

          BD=0.0
          MC=0.0

          DC 200 L=1,2
                  IF (DBRG(L).GT.180.0) DBRG(L)=DBRG(L)-360
          BRG(L)=DBRG(L)*DTOR
200       CONTINUE

          IF (TIME.LT.0) GOTO 800
```

78

```fortran
        DT=TIME-TIMEM1
        DTS(NP)=DT

        CALL FINDPHI(PHI,DT)

        XS1=XS(1)
        YS1=YS(1)
        XS2=XS(2)
        YS2=YS(2)
        BRG1=BRG(1)
        BRG2=BRG(2)

        CALL MP(XS1,YS1,XS2,YS2,BRG1,BRG2,ZX,ZY)

        IF(TIME.EQ.0) THEN
                CALL INIT(XS1,YS1,XS2,YS2,BRG1,BRG2,XKK,PKK)
        ENDIF

C *** PROJECT AHEAD STATE ESTIMATES ***
C      X(K+1/K) = PHI * X(K/K)

        CALL MATMUL(PHI,XKK,4,4,1,XKKM1)

C *** DERIVATION OF THE Q MATRIX ***

        CALL GETQ(DT,XKKM1,PKK(1,1),PKK(3,3),Q)

C *** PROJECT AHEAD ERROR COVARIANCE ESTIMATES ***
C        P(K+1/K) = (PHI * P(K/K) * PHIT) + Q

        CALL MATRAN(PHI,PHIT,4,4)
        CALL MATMUL(PHI,PKK,4,4,4,TEMP6)
        CALL MATMUL(TEMP6,PHIT,4,4,4,TEMP4)
301     CALL MATADD(TEMP4,Q,4,4,1,PKKM1)

C ***

        IF (TIME.EQ.0) THEN
           DO 542 I=1,4
            DO 542 J=1,4
             PKKM1(I,J)=0.0
542        CONTINUE

           PKKM1(1,1)=10000.0
           PKKM1(3,3)=9999.9
           PKKM1(2,2)=0.25
           PKKM1(4,4)=PKKM1(2,2)

           DO 543 I=1,4
            DO 543 J=1,4
             SPKKM1(I,J)=PKKM1(I,J)
543        CONTINUE
        ENDIF

        IF (MC.EQ.1.0) GOTO 303
```
79

```
C *** CALCULATE THE RESIDUAL DUE TO THE DIFFERENCE BETWEEN
C       OBSERVED AND ESTIMATED POSITIONS ***
C            | Z(K) - X(K/K-1) |

        PX=ZX-XKKM1(1,1)
        PY=ZY-XKKM1(3,1)
        PD=(PX**2)+(PY**2)
        PDIFF=SQRT(PD)

C *** CALCULATE THE MANEUVER GATE THRESHOLD VALUES ***

        CALL MANDET(TIME,PDIFF,XKKM1(1,1),XKKM1(3,1),PKKM1(1,1),
     *            PKKM1(3,3),PKKM1(1,3),XPL,YPL,XPU,YPU,TL,TU)
        DO 640 IE=1,37
          WRITE(4,*)XPL(IE),YPL(IE),XPU(IE),YPU(IE)
640     CONTINUE

C *** STORE THE MANEUVER GATE VALUES AND RESIDUAL DUE TO THE
C       POSITION DIFFERENCE ***

        TLS(NP)=TL
        TUS(NP)=TU
        DR(NP)=PDIFF

C *** MANEUVER DETECTION/DIVERGENCE ALGORITHM ***

        IF ((PDIFF.GE.TL).AND.(PDIFF.LE.TU)) THEN
          WRITE(*,*)'MANEUVER POSSIBILITY'
          MC=1.0
          ZT=ZT+1
          IF (ZT.GE.2.0) THEN
            IF (TIMEX.GT.TIME) THEN
              TIMEX=TIME
              TIMEXB(NP)=TIME
              TIMEXLO(NP)=TIMEXB(NP)-(1*DTS(NP))
              WRITE(5,1042)TIMEXB(NP),TIMEXLO(NP)
1042          FORMAT(2I4)
            ENDIF
            CALL MATSCL(2.0,Q,4,4,Q)
            TIMEP(NP)=TIME
            WRITE(6,1048)TIMEP(NP)
1048        FORMAT(I4)
            GOTO 301
          ENDIF
        ELSE
          ZT=0.0
          TIMEX=5000
        ENDIF

C *** RECOGNAZITION OF THE BAD OBSERVATIONS ***

        IF (PDIFF.GE.TU) THEN
          IF (BOC.NE.1.0)THEN
            BD=1.0
            BOC=1.0
```

```
          ELSE
              BOC=0.0
          ENDIF
        ELSE
          BOC=0.0
        ENDIF

303     MC=0.0

C ***

        IF (TIME.EQ.0.0) THEN
          DO 544 I=1,4
            DO 544 J=1,4
              PKKM1(I,J)=SPKKM1(I,J)
544       CONTINUE
        ENDIF


204     CONTINUE

        DO 210 L=1,2

C *** CALCULATE RANGE TO TARGET ***

              XDIFF=XKKM1(1,1)-XS(L)
              YDIFF=XKKM1(3,1)-YS(L)
              RANGE=SQRT(XDIFF**2+YDIFF**2)

C *** UPDATE H MATRIX WITH LATEST STATE ESTIMATES ***
C     AND CALCULATE MEASUREMENT ERROR ***

              H(1,1)=YDIFF/RANGE**2
              H(1,2)=0.0
              H(1,3)=-XDIFF/RANGE**2
              H(1,4)=0.0

              BRKKM1=ATAN2(XDIFF,YDIFF)
              E(L)=BRG(L)-BRKKM1

C *** COMPUTE KALMAN GAIN MATRIX ***
C     G=PKKM1*HT*(H*PKKM1*HT+R)**(-1)

              CALL MATRAN(H,HT,1,4)
              CALL MATMUL(H,PKKM1,1,4,4,TEMP1)
              CALL MATMUL(TEMP1,HT,1,4,1,TEMP2)
              VARE(L)=TEMP2(1,1)+R
              FAC1=1/VARE(L)
              CALL MATMUL(PKKM1,HT,4,4,1,TEMP3)
              CALL MATSCL(FAC1,TEMP3,4,1,G)

C *** COMPANSATION OF THE BAD OBSERVATIONS ***

        IF (BD.EQ.1.0) THEN
          DO 730 I=1,4
```

81

```fortran
                       G(I,1)=0.0
730           CONTINUE
          ENDIF

C ***

          IF (L.EQ.1) THEN
                    G11=G(1,1)
                    G13=G(3,1)
            ELSE
                    G21=G(1,1)
                    G23=G(3,1)
          ENDIF

C *** COMPUTE UPDATED ESTIMATE ***
C     X(K/K) = X(K/K-1) + G * E, WHERE E = Z(K) - H(K)*X(K/K-1)

                    XKK(1,1)=XKKM1(1,1)+(G(1,1)*E(L))
                    XKK(2,1)=XKKM1(2,1)+(G(2,1)*E(L))
                    XKK(3,1)=XKKM1(3,1)+(G(3,1)*E(L))
                    XKK(4,1)=XKKM1(4,1)+(G(4,1)*E(L))

C *** COMPUTE UPDATED ERROR COVARIANCE MATRIX ***
C     P(K/K) = (I - G*H) * P(K/K-1)

                    CALL MATMUL(G,H,4,1,4,TEMP4)
                    CALL MATSUB(IMAT,TEMP4,4,4,TEMP5)
                    CALL MATMUL(TEMP5,PKKM1,4,4,4,PKK)

C *** IF MORE MEASUREMENTS,***

                    IF (L.LT.2) THEN

C *** USE UPDATED STATE AND ERROR COVARIANCE ESTIMATES FOR NEXT
C     MEASUREMENT ***

                    DO 150 I=1,4
                    DO 150 J=1,4
                            PKKM1(I,J)=PKK(I,J)
                            XKKM1(I,1)=XKK(I,1)
150                 CONTINUE
                    ENDIF
210       CONTINUE


C *** THESE STATEMENTS ARE FOR THE SMOOTHING ALGORITHM ***

                    DO 620 I=1,4
                     XKKS(I,1,NP)=XKK(I,1)
620                 CONTINUE


                    DO 630 I=1,4
                     DO 630 J=1,4
                       PKKS(I,J,NP)=PKK(I,J)
630                 CONTINUE
```

82

```
C *** COMPUTE TRUE TRACKING AND OBSERVATION ERRORS ***

        TRKERR(NP)=SQRT((XT-XKK(1,1))**2+(YT-XKK(3,1))**2)
        OBSERR(NP)=SQRT((XT-ZX)**2+(YT-ZY)**2)

C *** COMPUTE ESTIMATED X-Y POSITION, COURSE, AND SPEED ***

        XPOS=XKK(1,1)
        YPOS=XKK(3,1)
        IF (XKK(2,1).EQ.0 .AND. XKK(4,1).EQ.0) THEN
                HDG=0.0
        ELSE
                HDG=RTOD*ATAN2(XKK(2,1),XKK(4,1))
        ENDIF
        IF (HDG.LT.0.0) HDG=HDG+360
        SPD(NP)=60*SQRT(XKK(2,1)**2+XKK(4,1)**2)

        WRITE(7,1011)TIME,XT,YT
1011    FORMAT(I4,2F15.4)
        WRITE(8,1012)TIME,NP,XPOS,YPOS,ZX,ZY,TRKERR(NP),OBSERR(NP),
     *                  PKK(1,1)
1012    FORMAT(2I4,7F15.4)

C *** UPDATE DATA COUNTER ***

        NP=NP+1
        TIMEM1=TIME

        GOTO 810

800     NP=NP-1


C**********************************************************************
C*     THIS IS WHERE THE FIXED-INTERVAL SMOOTHING ALGORITHM STARTS     *
C**********************************************************************

        WRITE(*,*)'FIXED-INTERVAL SMOOTHING NOW STARTS'
        WRITE(*,*)'***===========================***'

        DO 3000 KK=1,NP-1

         K=NP-KK

         DT=DTS(K+1)

         TIME=TIMEM1-DT
         CALL FINDPHI(PHI,DT)

         DO 901 I=1,4
          XSS(I,1)=XKKS(I,1,K)
901      CONTINUE

         DO 902 I=1,4
          DO 902 J=1,4
```

83

```
              PSS(I,J)=PKKS(I,J,K)
902           CONTINUE

C *** CALCULATE THE PREDICTED STATE ESTIMATES ***
C      X(K+1/K)=PHI*X(K/K)

           CALL MATMUL (PHI,XSS,4,4,1,XKKM1S)

C *** DERIVATION OF THE Q MATRIX ***

           CALL GETQ(DT,XKKM1S,PSS(1,1),PSS(3,3),Q)

C *** CALCULATION OF THE PREDICTED ERROR COVARIANCE MATRIX
C      AND COMPANSATION ALGORITHM WHICH USES THE MANEUVER
C      PERIOD DETECTED IN THE EXTENDED KALMAN FILTER ROUTINE ***
C      P(K+1/K)=PHI*P(K/K)*PHIT+Q

              CALL MATRAN (PHI,PHIT,4,4)
              CALL MATMUL(PHI,PSS,4,4,4,TEMP6)
              CALL MATMUL(TEMP6,PHIT,4,4,4,TEMP4)

              IF (TIME.EQ.TIMEP(K)) THEN
                   IF (TIME.EQ.TIMEP(1)) GOTO 483
                   CALL MATSCL(2.0,Q,4,4,Q)
              ELSE
305               READ(6,1051,END=482)TIMEM,TIMEL
1051              FORMAT(2I4)
                  IF (TIME.EQ.TIMEL) THEN
                       CALL MATSCL(2.0,Q,4,4,Q)
                  ENDIF
                  GOTO 305
              ENDIF
482           REWIND 6
483           CALL MATADD(TEMP4,Q,4,4,1,PKKM1S)

C *** CALCULATE THE SMOOTHING FILTER GAIN MATRIX ***
C      AK=P(K/K)*PHIT*INV⁰P(K+1/K)

              CALL MATINV (PKKM1S,4,IPKKM1S)
              CALL MATMUL (PHIT,IPKKM1S,4,4,4,TEMP1S)
              CALL MATMUL (PSS,TEMP1S,4,4,4,AK)


              DO 904 I=1,4
                 XNNM1(I,1)=XKKS(I,1,K+1)
904           CONTINUE


C *** CALCULATE THE SMOOTHED STATE ESTIMATE ***
C      XKKS=X(K/K)+AK*(X(K+1/N)-X(K+1/K)

              CALL MATSUB (XNNM1,XKKM1S,4,1,TEMP2S)
              CALL MATMUL (AK,TEMP2S,4,4,1,TEMP3S)
              CALL MATADD (XSS,TEMP3S,4,1,1,TH1)

              DO 903 I=1,4
```

```
                XKKS(I,1,K)=TH1(I,1)
903             CONTINUE

                DO 906 I=1,4
                 DO 906 J=1,4
                   PNNM1(I,J)=PKKS(I,J,K+1)
906             CONTINUE


C *** CALCULATE THE SMOOTHED COVARIANCE MATRIX ***
C       PKKS=P(K/K)+AK*|P(K+1/N)-P(K+1/K)|*AKT

                CALL MATSUB (PNNM1,PKKM1S,4,4,TEMP4S)
                CALL MATRAN (AK,AKT,4,4)
                CALL MATMUL (AK,TEMP4S,4,4,4,TEMP5S)
                CALL MATMUL (TEMP5S,AKT,4,4,4,TEMP6S)
                CALL MATADD (PSS,TEMP6S,4,4,1,TH2)

                DO 908 I=1,4
                 DO 908 J=1,4
                   PKKS(I,J,K)=TH2(I,J)
908             CONTINUE


C *** COMPUTE ESTIMATED X-Y POSITION, COURSE, AND SPEED ***

                IF (XKKS(2,1,K).EQ.0 .AND. XKKS(4,1,K).EQ.0) THEN
                        SHDG=0.0
                ELSE
                        SHDG=RTOD*ATAN2(XKKS(2,1,K),XKKS(4,1,K))
                ENDIF
                IF (SHDG.LT.0.0) SHDG=SHDG+360
                SSPD(K)=60*SQRT(XKKS(2,1,K)**2+XKKS(4,1,K)**2)

                TIMEM1=TIME

3000    CONTINUE

                REWIND 4

C *** CALCULATE THE SMOOTHED TRACKING ERROR ***

                DO 1100 K=1,NP
                   SXPOS=XKKS(1,1,K)
                   SYPOS=XKKS(3,1,K)
                   READ(4,1110)TIME,XT,YT
                   STRKERR(K)=SQRT((XT-XKKS(1,1,K))**2+(YT-XKKS(3,1,K))**2)

                   WRITE(9,1120)K,SXPOS,SYPOS,STRKERR(K),PKKS(1,1,K)

1100    CONTINUE
1110    FORMAT(I4,2F15.4)
1120    FORMAT(I4,4F20.4)

                CLOSE(UNIT=2)
                CLOSE(UNIT=3)
```

```
      CLOSE(UNIT=4)
      CLOSE(UNIT=5)
      CLOSE(UNIT=6)
      CLOSE(UNIT=7)
      CLOSE(UNIT=8)
      CLOSE(UNIT=9)

      WRITE(*,*)'THERE WERE',NP,'   OBSERVATIONS PROCESSED.'
      WRITE(*,*)'FOR GRAPHIC RESULTS COPY'
      WRITE(*,*)'  1) FILDATA.DAT'
      WRITE(*,*)'  2) MANEUDATA.DAT'
      WRITE(*,*)'  3) SMDATA.DAT'
      WRITE(*,*)'  4) TRUDATA.DAT'
      WRITE(*,*)'TO THE MATLAB SUB-DIRECTORY AND RUN  ==> <SHIPTR.M>'
      STOP
      END


C******************************************************************
C*                        SUBROUTINES                             *
C******************************************************************


      SUBROUTINE FINDPHI(PHI,DT)
C ****************************************************************
C      COMPUTES THE VALUES OF THE PHI MATRIX
C ****************************************************************
      REAL*4 PHI(4,4),DT

      DO 1501 I=1,4
      DO 1501 J=1,4
      DO 1501 K=1,2
            PHI(I,J)=0.0
1501  CONTINUE

C *** COMPUTE PHI MATRIX ***
      DO 1500 I=1,4
       PHI(I,I)=1.0
1500  CONTINUE
      PHI(1,2)=DT
      PHI(3,4)=DT

      RETURN

      END


      SUBROUTINE INIT(XS1,YS1,XS2,YS2,BRG1,BRG2,XKK,PKK)
C ****************************************************************
C      THIS ROUTINE INITIALIZES THE STATE
C      AND ERROR COVARIANCE ESTIMATES
C ****************************************************************
      REAL*4 XKK(4,1),PKK(4,4)
      REAL*4 XS1,YS1,XS2,YS2,BRG1,BRG2
      REAL*4 NUMER,DENOM
```

```
C *** INITIAL STATE ESTIMATE ***

        NUMER=(-YS2*TAN(BRG2))+(YS1*TAN(BRG1))+XS2-XS1
        DENOM=TAN(BRG1)-TAN(BRG2)

        XKK(3,1)=NUMER/DENOM
        XKK(2,1)=0.0
        XKK(1,1)=(XKK(3,1)-YS1)*TAN(BRG1)+XS1
        XKK(4,1)=0.0

C *** INITIAL ERROR COVARIANCE ESTIMATE ***

        DO 555 I=1,4
         DO 555 J=1,4
           PKK(I,J)=0.0
555     CONTINUE

        PKK(1,1)=10000.0
        PKK(3,3)=9999.9
        PKK(2,2)=0.25
        PKK(4,4)=PKK(2,2)

        RETURN

        END




        SUBROUTINE GETQ(DT,XKKM1L,P11,P33,Q)
C*******************************************************************************
C  CALCULATES STATE EXCITATION MATRIX Q
C*******************************************************************************
        REAL*4 DT,XKKM1L(4,1),QT,P11,P33,NT,Q(4,4)
        REAL*4 QM,VT,SIGTHT,SIGVT

        SIGTHT=0.0001
        SIGVT=0.0001

        IF ((XKKM1L(2,1).EQ.0).OR.(XKKM1L(4,1).EQ.0)) THEN
           QT=0.0
           GOTO 200
        ENDIF

        VT=SQRT((XKKM1L(2,1)**2)+(XKKM1L(4,1)**2))
        IF (P11.GT.P33) THEN
         QT=(((XKKM1L(2,1)/VT)**2)*SIGVT)+((XKKM1L(4,1)**2)*SIGTHT)
        ELSE
         QT=(((XKKM1L(4,1)/VT)**2)*SIGVT)+((XKKM1L(2,1)**2)*SIGTHT)
        ENDIF
200     NT=(DT**4)/4.0
        QM=NT*QT

        DO 556 I=1,4
         DO 556 J=1,4
          Q(I,J)=0.0
556     CONTINUE
```

```
         DO 557 I=1,4
          Q(I,I)=QM
557      CONTINUE
         CALL MATSCL(0.1,Q,4,4,Q)

         RETURN

         END



         SUBROUTINE MP(XS1,YS1,XS2,YS2,BRG1,BRG2,ZX,ZY)
C  *********************************************************************
C      THIS ROUTINE COMPUTES THE OBSERVED X, Y POSITIONS OF THE
C      TARGET USING SENSOR SHIP POSITIONS AND BEARINGS TO THE TARGET
C  *********************************************************************
         REAL*4 ZX,ZY
         REAL*4 XS1,YS1,XS2,YS2,BRG1,BRG2
         REAL*4 NUMER,DENOM

         NUMER=(-YS2*TAN(BRG2))+(YS1*TAN(BRG1))+XS2-XS1
         DENOM=TAN(BRG1)-TAN(BRG2)

         ZY=NUMER/DENOM
         ZX=(ZY-YS1)*TAN(BRG1)+XS1

         RETURN

         END

         SUBROUTINE MANDET(TIME,DIFF,XT,YT,P1,P3,P13,XPL,YPL,
     *                     XPU,YPU,TL,TU)
C  *********************************************************************
C      THIS SUBROUTINE COMPUTES THE THRESHOLD VALUES OF THE
C      MANEUVER GATES USING ERROR ELLIPSE EQUATIONS
C  *********************************************************************
         REAL*4 XT,YT,XPL(21),YPL(21),XPU(21),YPU(21),THE1,SIG2X
         REAL*4 SX,SY,CT,P1,P13,P3,DIFF,TL,TU,C,D,DTOR,A,B,SIG2Y
         REAL*4 THETA,DIV

         INTEGER*4 NP,TIME,CO

         DTOR=0.0174529
         DIV=30.0
         A=2*P13
         B=P1-P3
         THE1=0.5*ATAN2(A,B)
         C=(P1+P3)/2
         D=0.0
         IF (P13.EQ.0.0) GOTO 10
         D=P13/SIN(2.0*THE1)
10       SIG2X=ABS(C+D)
         SIG2Y=ABS(C-D)
         SX=(SIG2X**0.5)
         SY=(SIG2Y**0.5)
         IF (SX.GT.SY) THEN
             TL=3*SX
```

```
            TU=8*SX
        ELSE
            TL=3*SY
            TU=8*SY
        ENDIF
        CT=COS(THE1)
        ST=SIN(THE1)

        IF (TIME.GT.0) THEN
            TL=TL/DIV
            TU=TU/DIV
        ENDIF

        WRITE(3,1045)TIME,TL,TU,DIFF
1045    FORMAT(I4,3F10.4)

        DO 100 IE=1,37
                CO=IE-1
                THETA=((360/36)*CO)*DTOR
                XPU(IE)=TU*COS(THETA)+XT
                YPU(IE)=TU*SIN(THETA)+YT
100     CONTINUE

        DO 120 IE=1,37
                CO=IE-1
                THETA=((360/36)*CO)*DTOR
                XPL(IE)=TL*COS(THETA)+XT
                YPL(IE)=TL*SIN(THETA)+YT
120     CONTINUE
        RETURN

        END




        SUBROUTINE MATMUL(A,B,L,M,N,C)
C ***************************************************************
C       THIS ROUTINE MULTIPLIES TWO MATRICES TOGETHER
C       ⁰ C(L,N) = A(L,M) * B(M,N)
C ***************************************************************
        REAL*4 A(L,M),B(M,N),C(L,N)

        DO 10 I=1,L
        DO 10 J=1,N
         C(I,J)=0.0
10      CONTINUE

        DO 100 I= 1,L
        DO 100 J= 1,N
        DO 100 K= 1,M
         C(I,J) = C(I,J) + A(I,K)*B(K,J)
100     CONTINUE

        RETURN

        END
```

```
      SUBROUTINE MATRAN(A,B,N,M)
C ***************************************************************
C     THIS ROUTINE TRANSPOSES A MATRIX
C           ° B(M,N) = A'(N,M)
C ***************************************************************
      REAL*4 A(N,M), B(M,N)

      DO 100 I= 1,N
      DO 100 J= 1,M
       B(J,I) = A(I,J)
100   CONTINUE

      RETURN

      END


      SUBROUTINE MATSCL(SC,A,N,M,C)
C ***************************************************************
C     THIS ROUTINE MULTIPLIES A MATRIX WITH A SCALAR
C     ° C(N,M) = SC * A(N,M)
C ***************************************************************
      REAL*4 A(N,M), C(N,M), SC

      DO 100 I = 1,N
      DO 100 J = 1,M
       C(I,J) = SC*A(I,J)
100   CONTINUE

      RETURN

      END


      SUBROUTINE MATSUB(A,B,N,M,C)
C ***************************************************************
C     THIS ROUTINE SUBTRACTS TWO MATRICES
C     ° C(N,M) = A(N,M) - B(N,M)
C ***************************************************************
      REAL*4  A(N,M),B(N,M),C(N,M)

      DO 100 I = 1,N
      DO 100 J = 1,M
       C(I,J)=A(I,J)-B(I,J)
100   CONTINUE

      RETURN

      END

      SUBROUTINE MATADD(A,B,N,M,L,C)
C***************************************************************
C     THIS ROUTINE ADDS TWO MATRICES
C     ° C(N,M) = A(N,M) + B(N,M)
C***************************************************************
```

```fortran
        REAL*4  A(N,M),B(N,M),C(N,M,L)
        DO 100 I = 1,N
        DO 100 J = 1,M
         C(I,J,L)=A(I,J)+B(I,J)
100     CONTINUE

        RETURN
        END


        SUBROUTINE MATINV (A,N,C)
C***************************************************************
C       THIS ROUTINE COMPUTES THE INVERSE OF
C       A MATRIX
C           C(N,N)=INV |A(N,N)|
C***************************************************************
        REAL*4 A(N,N),C(N,N),D(20,20)
        DO 100 I = 1,N
         DO 100 J = 1,N
100          D(I,J)=A(I,J)

        DO 115 I=1,N
         DO 115 J=N+1,2*N
115     D(I,J)=0.0

        DO 120 I=1,N
         J=I+N
120     D(I,J)=1.0

        DO 240 K=1,N
         M=K+1
         IF (K.EQ.N) GOTO 180
         L=K
         DO 140 I=M,N
140      IF (ABS(D(I,K)).GT.ABS(D(L,K))) L=I
         IF (L.EQ.K) GOTO 180

         DO 160 J=K,2*N
          TEMP=D(K,J)
          D(K,J)=D(L,J)
160      D(L,J)=TEMP

180      DO 185 J=M,2*N
185      D(K,J)=D(K,J)/D(K,K)

         IF (K.EQ.1) GOTO 220
         M1=K-1
         DO 200 I=1,M1
          DO 200 J=M,2*N
200     D(I,J)=D(I,J)-D(I,K)*D(K,J)

         IF (K.EQ.N) GOTO 260

220      DO 240 I=M,N
          DO 240 J=M,2*N
240         D(I,J)=D(I,J)-D(I,K)*D(K,J)
```

```
260      DO 265 I=1,N
          DO 265 J=1,N
           K=J+N
265      C(I,J)=D(I,K)

         RETURN
         END
```

APPENDIX B.   INPUT DATA FILE FORMATTING ALGORITHM

```
C *** RAWDATA.FOR ***

C*************************************************************************
C*                                                                      *
C*  THIS PROGRAM EMPLOYES A TARGET AND TWO SENSOR SHIPS.  IT ASKS FOR   *
C* THE INITIAL POSITIONS, SPEEDS AND COURSES OF THE TARGET AND SENSOR   *
C* SHIPS. IT ALSO CALLS FOR MANEUVER PERIOD, THE SPEED AND THE COURSE   *
C* CHANGE OF THE TARGET DURING THIS MANEUVER PERIOD.  THE OUTPUTS       *
C* WHICH CONSIST OF NOISY OR NOISE FREE BEARINGS FROM THE SENSOR SHIPS  *
C* TO THE TARGET AND POSITIONS OF THE EACH SHIPS ARE STORED IN THE FILE *
C* TRKDATA.DAT TO BE USED BY THE PROGRAM <SHIPMANE.FOR>.                *
C*************************************************************************

C *** VARIABLE DEFINATIONS ***

C      BRG            = MEASURED TARGET BEARINGS.
C      CASE           = INDICATOR OF THE NOISE EXISTANCE.  THE NOISE EXISTS
C                       FOR POSITIVE VALUE, NO NOISE FOR NEGATIVE VALUE.
C      CS, CE         = START AND END HEADINGS OF THE MANEUVER.
C      DTOR           = DEGREE TO RADIAN CONVERSITION FACTOR.
C      END            = END OF THE TRACKING PROBLEM.
C      HDGS           = SENSOR SHIP'S HEADING.
C      HDGT           = TARGET'S HEADING.
C      HDGTD          = TOTAL HEADING CHANGE DURING THE MANEUVER.
C      MS, ME         = START AND END TIMES OF THE MANEUVER.
C      N1, N2         = MEASUREMENT NOISESS.
C      NM             = NUMBER OF MANEUVERS.
C      PER            = OBSERVATION PERIOD.
C      RTOD           = RADIAN TO DEGREE CONVERSITION FACTOR.
C      SS, SE         = START AND END TIMES OF THE MANEUVER.
C      SPDS           = SENSOR SHIP'S SPEED.
C      SPDT           = TARGET'S SPEED.
C      SPDTD          = TOTAL SPEED CHANGE DURING THE MANEUVER.
C      TIMED          = TOTAL MANEUVER TIME.
C      UNHDGCH        = HEADING CHANGE PER OBSERVATION.
C      UNSPDCH        = SPEED CHANCE PER OBSERVATION.
C      XDIFF, YDIFF   = THE DISTANCES IN THE X AND Y DIRECTIONS FROM SENSOR
C                       TO A TARGET POSITION.
C      XS             = SENSOR SHIP'S STATES.
C      XT             = TARGET'S STATES.

C *** VARIABLE DECLERATIONS ***

       REAL*4 XT(4,1),XS1(4,1),PHI(4,4),SPDS1,HDGS1,SPDS2,HDGS2,SP,HD
       REAL*4 DT,SPDT,HDGT,XS2(4,1),TEMP1(4,1),CASE,XDIFF1,YDIFF1
       REAL*4 XDIFF2,YDIFF2,N1,N2,DTOR,RTOD,BRG1,BRG2,CS,CE(20),HDGTD
       REAL*4 MS(20),ME(20),SS,SE(20),SPDTD,UNSPDCH(10),UNHDGCH(10)

       INTEGER TIME,TIMEM1,NM,PER,END

C *** OPEN DATA FILES ***
```

93

```fortran
      OPEN(UNIT=2,FILE='NOISE1.DAT',STATUS='OLD')
      OPEN(UNIT=3,FILE='NOISE2.DAT',STATUS='OLD')
      OPEN(UNIT=4,FILE='TRKDATA.DAT',STATUS='NEW')

C **************************************************************

      WRITE(*,*)'ENTER A NEGATIVE NUMBER FOR NOISELESS CASE;'
      WRITE(*,*)'POSITIVE FOR NOISY CASE'
      READ(*,*)CASE

      TIMEM1=0
      RTOD=57.29577951
      DTOR=0.017453293

      WRITE(*,*)'ENTER THE OBSERVATION PERIOD AND'
      WRITE(*,*)'END OF THE OBSERVATION TIME.'
      READ(*,*)PER,END

      WRITE(*,*)'INPUT DESIRED INITIAL X POSITION, Y POSITION,'
      WRITE(*,*)'SPEED (IN KNOTS) AND COURSE (IN DEGREES) OF TARGET'
      READ(*,*)XT(1,1),XT(3,1),SPDT,HDGT

      SP=SPDT
      HD=HDGT
      XT(2,1)=(SPDT/60)*SIN(HDGT*DTOR)
      XT(4,1)=(SPDT/60)*COS(HDGT*DTOR)

      WRITE(*,*)'FOR SENSOR 1:'
      WRITE(*,*)'INPUT DESIRED INITIAL X POSITION, Y POSITION,'
      WRITE(*,*)'SPEED (IN KNOTS) AND COURSE (IN DEGREES)'
      READ(*,*)XS1(1,1),XS1(3,1),SPDS1,HDGS1

      XS1(2,1)=(SPDS1/60)*SIN(HDGS1*DTOR)
      XS1(4,1)=(SPDS1/60)*COS(HDGS1*DTOR)

      WRITE(*,*)'FOR SENSOR 2:'
      WRITE(*,*)'INPUT DESIRED INITIAL X POSITION, Y POSITION,'
      WRITE(*,*)'SPEED (IN KNOTS) AND COURSE (IN DEGREES)'
      READ(*,*)XS2(1,1),XS2(3,1),SPDS2,HDGS2

      XS2(2,1)=(SPDS2/60)*SIN(HDGS2*DTOR)
      XS2(4,1)=(SPDS2/60)*COS(HDGS2*DTOR)

      WRITE(*,*)'HOW MANY TIMES DO YOU WANT TO MAKE MANEUVER?'
      READ(*,*)NM

      DO 540 K=1,NM
      WRITE(*,*)' '
      WRITE(*,*)'**MANEUVER #',K

510   WRITE(*,*)'ENTER THE STARTING AND ENDING TIMES OF'
      WRITE(*,*)'THE MANEUVER #',K
      READ(*,*)MS(K),ME(K)

      IF ((MS(K).GT.END).OR.(ME(K).GT.END)) THEN
```

```fortran
          WRITE(*,*)'CAREFULL! END OF THE TRACKING IS',END
          WRITE(*,*)' '
          GOTO 510
       ENDIF

       TIMED=ME(K)-MS(K)

520    WRITE(*,*)'ENTER THE STARTING AND ENDING SPEEDS OF'
       WRITE(*,*)'THE SPEED MANEUVER #',K
       READ(*,*)SS,SE(K)

       IF (SS.NE.SP) THEN
          WRITE(*,*)'CAREFULL! CURRENT SPEED IS',SP
          WRITE(*,*)' '
          GOTO 520
       ENDIF

       SP=SE(K)
       SPDTD=SE(K)-SS
       UNSPDCH(K)=(SPDTD/TIMED)*PER

530    WRITE(*,*)'ENTER THE STARTING AND ENDING COURSES OF'
       WRITE(*,*)'THE COURSE MANEUVER #',K
       READ(*,*)CS,CE(K)

       IF (CS.NE.HD) THEN
          WRITE(*,*)'CAREFUL! CURRENT HEADING IS',HD
          WRITE(*,*)' '
          GOTO 530
       ENDIF

       HD=CE(K)
       HDGTD=CE(K)-CS
       UNHDGCH(K)=(HDGTD/TIMED)*PER

540    CONTINUE

       DO 610 J=1, 1000
          DO 550 L=1,NM

             IF ((TIME.GT.(MS(L))).AND.(TIME.LE.(ME(L)))) THEN

                SPDT=SPDT+UNSPDCH(L)
                HDGT=HDGT+UNHDGCH(L)

                IF ((UNSPDCH(L).LT.0.0).AND.(SPDT.LT.SE(L))) SPDT=SE(L)
                IF ((UNSPDCH(L).GT.0.0).AND.(SPDT.GT.SE(L))) SPDT=SE(L)

                IF ((UNHDGCH(L).LT.0.0).AND.(HDGT.LT.CE(L))) HDGT=CE(L)
                IF ((UNHDGCH(L).GT.0.0).AND.(HDGT.GT.CE(L))) HDGT=CE(L)

                XT(2,1)=(SPDT/60)*SIN(HDGT*DTOR)
                XT(4,1)=(SPDT/60)*COS(HDGT*DTOR)

             ENDIF
```

```fortran
550        CONTINUE

C *** UPDATE TARGET AND SENSOR STATES TO MEASUREMENT TIME ***

           DT=TIME-TIMEM1

C *** COMPUTE PHI MATRIX ***

           CALL FINDPHI(DT,PHI)

C *** UPDATE TARGET STATES ***
           CALL MATMUL(PHI,XT,4,4,1,TEMP1)
           DO 560 I=1,4
              XT(I,1)=TEMP1(I,1)
560        CONTINUE

C *** UPDATE SENSOR STATES ***

           CALL MATMUL(PHI,XS1,4,4,1,TEMP1)
           DO 570 I=1,4
              XS1(I,1)=TEMP1(I,1)
570        CONTINUE

           CALL MATMUL(PHI,XS2,4,4,1,TEMP1)
           DO 580 I=1,4
              XS2(I,1)=TEMP1(I,1)
580        CONTINUE

           XDIFF1=XT(1,1)-XS1(1,1)
           YDIFF1=XT(3,1)-XS1(3,1)

           XDIFF2=XT(1,1)-XS2(1,1)
           YDIFF2=XT(3,1)-XS2(3,1)

           READ(2,*)N1
           READ(3,*)N2

           IF (CASE.GE.0.0) GOTO 590
              N1=0.0
              N2=0.0

590        BRG1=RTOD*ATAN2(XDIFF1,YDIFF1)+N1
           IF (BRG1.LT.0.0) BRG1=BRG1+360
           BRG2=RTOD*ATAN2(XDIFF2,YDIFF2)+N2
           IF (BRG2.LT.0.0) BRG2=BRG2+360

           WRITE(4,600)TIME,XT(1,1),XT(3,1),XS1(1,1),XS1(3,1),
     *                 BRG1,XS2(1,1),XS2(3,1),BRG2
600        FORMAT(I4,8F9.4)

           TIMEM1=TIME
           TIME=TIME+PER


       IF (TIME.GT.END) GOTO 620
```

96

```
610       CONTINUE

620       STOP
          END

          SUBROUTINE FINDPHI(DT,PHI)
C***********************************************************************
C    THIS ROUTINE COMPUTES PHI MATRIX
C***********************************************************************
C         DIMENSIONS AND DECLERATIONS
          REAL*4 PHI(4,4),DT

          PHI(1,1)=1.0
          PHI(1,2)=DT
          PHI(1,3)=0.0
          PHI(1,4)=0.0
          PHI(2,1)=0.0
          PHI(2,2)=1.0
          PHI(2,3)=0.0
          PHI(2,4)=0.0
          PHI(3,1)=0.0
          PHI(3,2)=0.0
          PHI(3,3)=1.0
          PHI(3,4)=DT
          PHI(4,1)=0.0
          PHI(4,2)=0.0
          PHI(4,3)=0.0
          PHI(4,4)=1.0

          RETURN

          END

          SUBROUTINE MATMUL(A,B,L,M,N,C)
C    **********************************************************************
C         THIS ROUTINE MULTIPLIES TWO MATRICES TOGETHER
C         ⁰ C(L,N) = A(L,M) * B(M,N)
C    **********************************************************************
C         DIMENSIONS AND DECLARATIONS
          REAL*4 A(L,M),B(M,N),C(L,N)

          DO 10 I=1,L
          DO 10 J=1,N
           C(I,J)=0.0
10        CONTINUE

          DO 100 I= 1,L
          DO 100 J= 1,N
          DO 100 K= 1,M
           C(I,J) = C(I,J) + A(I,K)*B(K,J)
100       CONTINUE

          RETURN

          END
```

LIST OF REFERENCES

1. Kalman, R. E., *A New Approach to Linear Filtering and Prediction Problems*, Trans. ASME, J. Basic Eng., Vol. 82D, No.1, March 1960.

2. Kalman, R. E. and Bucy, R. S., *A New Results in Linear Filtering and Prediction Theory*, Trans. ASME, J. Basic Eng., Vol. 83D, No.1, March 1961.

3. Bennet, T. K., *Development of the Adaptive Extended Kalman Filter Ship-Tracking Algorithm Using Bearings-Only Measurements*, Master's Thesis, Naval Postgraduate School, June 1988.

4. Galinis, W. J., *Fixed Interval Smoothing Algorithm for an Extended Kalman Filter for Over-the-Horizon Ship Tracking*, Master's Thesis, Naval Postgraduate School, March 1989.

5. Meditch, J. S., *Stochastic Optimal Linear Estimation and Control*, McGraw-Hill, Inc., 1969.

6. Dittmar, C. A. Jr, *The Application of Extended Kalman Filtering to the Position Locating Reporting System (PLRS)*, Master's Thesis, Naval Postgraduate School, December 1975.

7. Brown, R. G., *Introduction to Random Signal Analysis and Kalman Filtering*, John Wiley & Sons, Inc., 1983.

8. Gelb, A., *Applied Optimal Estimation*, M. I. T. Press, 1974.

9. Maybeck, P. S., *Stochastic Models, Estimation, and Control Volume1*, Academic Press, 1979.

10. Spehn, S. L., *Noise Adaptation and Correlated Maneuver Gating of an Extended Kalman Filter*, Ph.D. Dissertation, Naval Postgraduate School, March 1990.

11. Maybeck, P. S., *Stochastic Models, Estimation, and Control Volume2*, Academic Press, 1982.

12. Ohanian, H. C., *Physics*, W. W. Norton & Company, nc., 1982.

13. Karaman, S., *Fixed Point Smoothing Algorithm to the Torpedo Tracking Problem*, Master's Thesis, Naval Postgraduate School, June 1986.

14. Olcovich, *Passive Acoustic Target Motion Analysis*, Master's Thesis, Naval Postgraduate School, June 1986.

INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Technical Information Center 2
   Cameron Station
   Alexandria, VA 22304-6145

2. Library, Code 52 2
   Naval Postgraduate School
   Monterey, CA 93943-5002

3. Deniz Kuvvetleri K.ligi 1
   Personel Sube Bsk.ligi
   Bakanliklar, Ankara / TURKEY

4. Kara Harp Okulu K.ligi 1
   Kutuphanesi
   Bakanliklar, Ankara / TURKEY

5. Deniz Harp Okulu K.ligi 1
   Kutuphanesi
   Tuzla, Istanbul / TURKEY

6. Hava Harp Okulu K.ligi 1
   Kutuphanesi
   Yesilyurt, Istanbul / TURKEY

7. Orta Dogu Teknik Universitesi 1
   Okul Kutuphanesi
   Balgat, Ankara / TURKEY

8. Bogazici Universitesi 1
   Okul Kutuphanesi
   Bebek, Istanbul / TURKEY

9. Ege Dokuz Eylul Universitesi 1
   Okul Kutuphanesi
   Cumhuriyet Meydani, Izmir / TURKEY

10. Chairman, Code EC 1
    Department of Electrical and Computer Engineering
    Naval Postgraduate School
    Monterey, CA 93943-5002

11. Professor H. A. Titus, Code EC/Ts 1
    Department of Electrical and Computer Engineering
    Naval Postgraduate School
    Monterey, CA 93943-5002

12. Assistant Professor J. Burl, Code EC/Bl 1
    Department of Electrical and Computer Engineering
    Naval Postgraduate School
    Monterey, CA 93943-5002

13. LTJG Alaettin SEVIM 1
    Inonu caddesi 250/1 Daire 25
    Hatay 35280 Izmir / TURKEY